DOCUMENT RESUME

ED 347 967 IR 014 241

AUTHOR Li, Zhongmin; Merrill, M. David
TITLE Implementation of an Expert System for Instructional Design: Phase 2. Knowledge Engineering Report.
INSTITUTION Human Technology, Inc., McLean, VA.; Office of Personnel Management, Washington, D.C.
SPONS AGENCY Army Research Inst. for the Behavioral and Social Sciences, Alexandria, Va.
PUB DATE 8 May 88
NOTE 82p.; For related reports, see IR 014 239-240. Reports on Phase 1 are not available.
PUB TYPE Reports - Research/Technical (143)

EDRS PRICE MF01/PC04 Plus Postage.
DESCRIPTORS *Computer System Design; *Expert Systems; *Instructional Design; Instructional Development; *Material Development; Military Training; Models; *Task Analysis; *Testing
IDENTIFIERS Elaboration Theory; *ID Expert System; Knowledge Representation

ABSTRACT

Instructional Design Expert (ID Expert) is a prototype instructional design expert system which supports the instructional design phase of instructional system development. It differs from other instructional design toolkits in that it is built using a knowledge engineering approach, and has been implemented as an expert system. The system incorporates both "public" knowledge (in the forms of instructional design theories and models) and "private" knowledge (representing the artistic aspect of instructional design). The project spelled out such "private" knowledge explicitly. This report presents the knowledge representation mechanisms used in ID Expert and related implementation issues. It identifies the five main stages in the development of an expert system (problem identification, conceptualization, formalization, implementation, and testing). The problem identification phase of ID Expert development was performed by senior knowledge engineers from Teknowledge, Inc. The focus of this report is on the remaining four stages. An introduction to the project is provided in the first section, and the second presents a conceptual model of instructional design implemented in ID Expert. Section three presents the results of the conceptualization and formalization phases, and section four describes the results of the implementation and testing phases. (30 references) (GL)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)
☒ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.
• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

ED347967

KNOWLEDGE ENGINEERING REPORT

Implementation of an Expert System for Instructional Design: Phase 2

Developed by:

Zhongmin Li
University of Southern California
and
M. David Merrill
Utah State University

with
Human Technology Inc.
McLean, Virginia
and
U.S. Office of Personnel Management
Office of Training and Development
Training Assistance Branch

Developed for:

Army Research Institute
for the Behavioral and Social Sciences
Alexandria, Virginia

May 8, 1988

IR014341

BEST COPY AVAILABLE

Table of Contents

List of Figures ii
List of Tables iv
Introduction 1
A Model of Instructional Design 3
Components of an Instructional System 3
Instructional Design Model Simulated in ID Expert 5
Knowledge Representation 8
Knowledge Representation Strategies 8
Mapping Instructional Design Model to Frame System 12
Content Structure Representation 15
Representing concept Taxonomy 15
Representing Procedural Hierarchy 24
Course Organization Representation 27
Transaction Representation 32
Implementation 32
Stages of Implementation of Related Tasks 33
Implementing Frame-Based System in S.1 37
Reasoning Strategies 41
Forward Reasoning 41
Backward Reasoning 45
Task-specific Reasoning Control 47
Reasoning with Certainty Factor 50
Nonmonotonic Reasoning 52
User Interface 56
Text Representation of Graphic Output 56
Help Facilities 57
Test and Evaluation 58
Aspects of ID Expert Being Evaluated 58
Methods of Evaluation 59
Reference 61

List of Figures

Figure 1.1. Development Phases for Building Expert Systems.....2
Figure 2.1. Components of an Instructional System.....4
Figure 2.2. The Instructional Design Model Simulated by ID Expert. 6
Figure 3.1. Major Objects and Their Relationships in the Instructional Design Model.....12
Figure 3.2. Module Frame.....13
Figure 3.3. Categories of Module Function.....15
Figure 3.4. A Concept Taxonomy for measures of Central Tendency.16
Figure 3.5. Representation of a Concept Taxonomy.....16
Figure 3.6. A Concept Tree and Its Corresponding Coordinate Set Tree.....17
Figure 3.7. Frames for Concept Taxonomy.....18
Figure 3.8. Coordinate Set Focuses and Their Associated Instructional Objective Patterns.....20
Figure 3.9. Dependency Between Coordinate Set Relationship and Coordinate Set Focus.....21
Figure 3.10. Module Types and Sample Rules for Their Selection.....22
Figure 3.11. Conflict Resolution Rules for Module Type Determination.....23
Figure 3.12. Integrative Modules for a Concept Taxonomy.....23
Figure 3.13. A Procedural Hierarchy for Spreadsheet Application.....24
Figure 3.14. Mechanism for Representing Procedural Hierarchy.....25
Figure 3.15. Termination Rules for Analyzing a Procedure.....26
Figure 3.16. Comparisons of Course Structure in Conventional Instruction and ID Expert.....27
Figure 3.17. Course Organization Element Frame.....28
Figure 3.18. Instructional Map Representation.....28
Figure 3.19. Sequence Procedure for Concept Taxonomy.....30
Figure 3.20. Course Organization Diagrams for Spreadsheet Budget. 31
Figure 3.21. Transaction Representation.....32
Figure 4.1. Interactions between a Domain Expert and a Knowledge Engineer at the Implementation Phase.....33
Figure 4.2. A Transaction Structure.....36
Figure 4.3. Partial Implementation of the MODULE Frame Using S.1 Objects.....38
Figure 4.4. Relationships between Classes and Their Class Type.....39
Figure 4.5. Examples of Rule Definition in S.1.....40

Figure 4.6. An Example of Control Block.....41
Figure 4.7. Instructional Design Problem Solver.....42
Figure 4.8. Use Control Block to Code Forward Reasoning Strategy.43

Figure 4.9. Forward Reasoning using Control Blocks and Invoke Statement. 45
Figure 4.10. An Inference Tree Generated by Backward Reasoning Strategy. 47
Figure 4.11. A Portion of a Dialogue with ID Expert to Determine the Level of Motivation 49
Figure 4.12. Rules with Certainty Factors. 50
Figure 4.13. Sample Output of Evidence Accumulation Using Certainty Factor 52
Figure 4.14. Dynamic Specification of Defaults 53
Figure 4.15. Selecting Highest Certainty Value as Defaults 54
Figure 4.16. Reconsideration Process in ID Expert 55
Figure 4.17. A Transcript of Reconsideration dialogue. 56
Figure 4.18. Text Presentation of Content Structure 56
Figure 4.19. Text Representation of Course Organization. 57
Figure 4.20. A Help Message Generated by ID Expert, which Includes Examples, and Definitions of Key Terms 58

List of Tables

Table 4.1. List of Additive Knowledge Bases of ID Expert.....34
Table 4.2. Additive Knowledge Bases Contained in Each Task Group.36

KNOWLEDGE ENGINEERING REPORT

Implementation of an Expert System for Instructional Design: Phase 2

Zhongmin Li
University of Southern California

M. David Merrill
Utah State University

1. Introduction

ID Expert (Instructional Design Expert System) is a prototype instructional design expert system which supports the instructional design phase of ISD (Instructional System Development). It differs from other instructional design toolkits in that it is built using a knowledge engineering approach, and has been implemented as an expert system. Hayes-Roth, Waterman, & Lenal (1983) describe two kinds of expert knowledge: "public" and "private":

> Public knowledge includes the published definitions, facts, and theories of which textbooks and references in the domain of study are typically composed. But expertise usually involves more than just this public knowledge. Human experts generally possess private knowledge that has not found its way into the published literature. This private knowledge consists largely of rules of thumbs that have come to be called heuristics. Heuristics enable the human expert to make educated guesses when necessary, to recognize promising approaches to problems, and to deal effectively with errorful or incomplete data. (p. 4)

In instructional design the public knowledge appears in the forms of instructional design theories and models. It is the "private" knowledge that represents the artistic aspect of instructional design. This project spelled out such "private" knowledge explicitly. In this report, we present the knowledge representation mechanisms used in ID Expert, and related

implementation issues. For an instructional design view of ID Expert see Merrill & Li (1988).

Since instructional design is a complex process, available resources prohibit the development of a full-scale system that can solve every instructional design problem. technologically, today's state of the art is that expert systems may solve simple problems easily and reliably (Winston, 1984). This project emphasized two small design tasks: (a) design a course based on a KINDS TAXONOMY content structure for concept classification, and (b) design a course based on PATH ALGORITHM content structure for device operation procedures. It was hoped that the methodologies developed, and formalization used in developing the prototype system could be generalized in developing expert systems for other instructional design tasks.

The development of an expert system consists of five main stages: (a) identification, (b) conceptualization, (c) formalization, (d) implementation, and (e) testing.

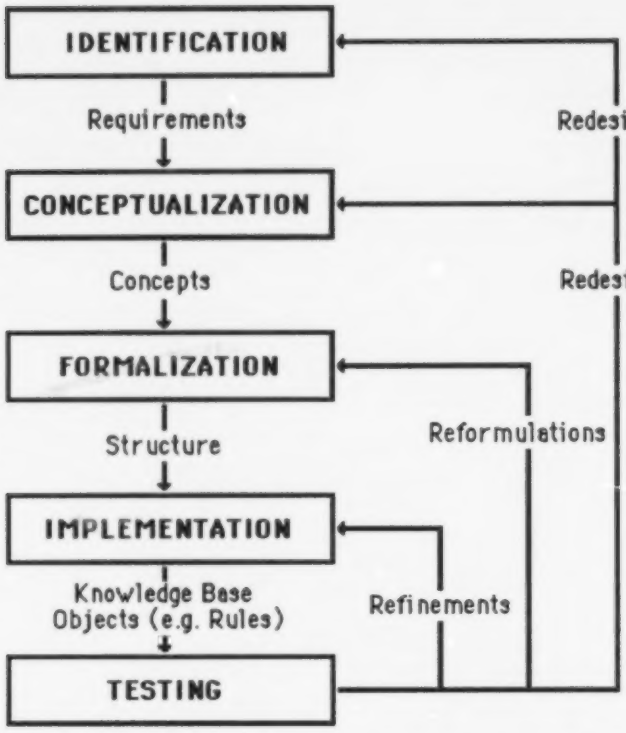


Figure 1.1. Development Phases for Building Expert Systems.

The identification stage is to characterize the important aspects of the problem under consideration. The conceptualization stage focuses on the explicit identification of key concepts, relations, and control mechanisms needed to describe the domain. Along the way, subtasks, strategies, and constraints related to the problem-solving activities are explored. The formalization stage involves the formal representation of the concepts and relations in the domain based on selected knowledge-engineering tools and methods. The implementation stage includes the construction of the computer program that utilizes the formalized knowledge. Finally, the testing stage involves evaluating the performance and utility of the program (Buchanan, Barstow, Bechtal, Bennett, Clancey, Kulikowski, Mitchell, & Waterman, 1983; Waterman, 1985).

The problem identification phase of ID Expert development was performed by senior knowledge engineers from Teknowledge, Inc. and the results were presented in a report by Human Technology, Inc. (1987). In this report we focus on the other four stages of ID Expert development. Section two presents a conceptual model of instructional design implemented in ID Expert. Section three presents results of

conceptualization and formalization phases, and Section four presents results of implementation and testing phases. Although the formalization phase is usually tied to a knowledge representation framework incorporated in an expert systems building language (Waterman, 1985), attempts were made to separate the knowledge representation from the language used, so that the knowledge base would be easily ported to other expert system building languages.

2. A Model of Instructional Design

The output of ID Expert is a design specification for an instructional system in selected domain. This section first describes six major components of an instructional system, and then presents the design activities that specify these components of the instructional system.

2.1. Components of an Instructional System

The instructional system conceptualized in ID Expert consists of six components: (a) course organization, (b) module, (c) transaction, (d) strategy, (e) content representation, and (f) sequence (Figure 2.1).

Content structure is the organization of the subject matter content. A content structure diagram is often included as part of the content representation. different content structures are appropriate for different instructional goals, and may require different instructional design procedures. The determination of content structure for a given subject matter decreases an instructional designer's searching spacing of instructional design procedures, thus increase the efficiency of the design.

Module is the fundamental unit of course organization specified by ID Expert. The module consists of three standard instructional components: (a) content representation of subject matter, (b) a set of transactions, and (c) a set of strategy rules as shown in Figure 2.1.

A transaction defines a mode of interaction between students and the content representation of the instructional materials. It consists of a sequence of presentations and reactions necessary to accomplish some instructional functions. It is a mutual, dynamic, real-time give-and-take between program and student in which they exchange information.

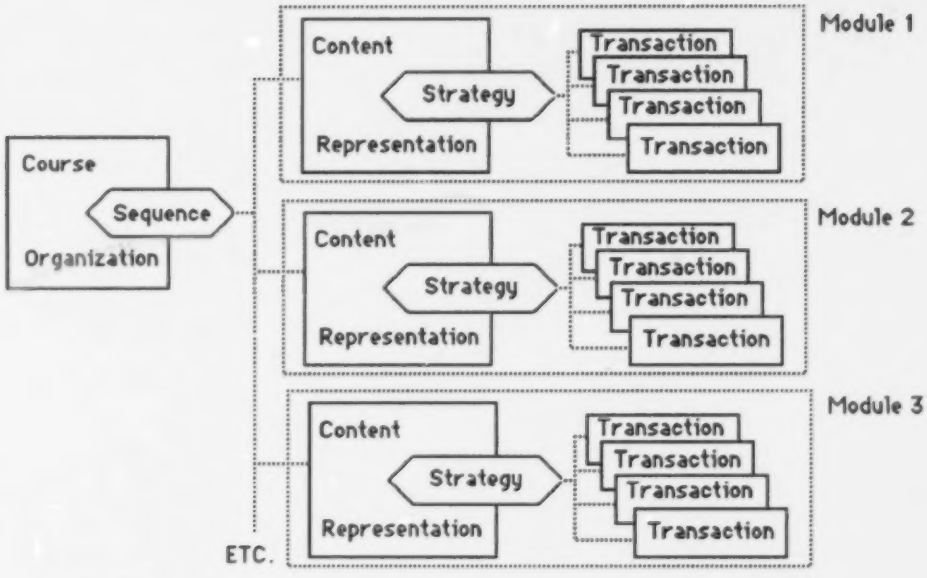


Figure 2.1. Components of an Instructional System.

Strategy is a set of rules for determining which transaction should be taught next, and who should make the decision during instructional delivery. The strategy design is out of the scope of the prototype ID Expert.

Content representation is the instructional materials with which students interact during instructional delivery.

Course organization is the module network and the possible paths that the student may take in traversing this network. The components of course organization are pre- and post integrative modules, instructional modules, menus. A course organization can be a component of another course organization. This nesting of course organizations defines the sequence of instructional units.

Sequence is a set of rules for determining which modules should be taught next, and who should make the decision. Sequence rules con be either static or dynamic. Static rules are used in instructional design phase to prescribe the sequence of instructional materials. Dynamic rules are used in instructional delivery phase to select next instructional topic based on the performance of students. For the prototype ID Expert, only static rules are implemented as a part of sequence algorithms for course organization.

2.2. Instructional Design Model Simulated in ID Expert

Reigeluth (1983) defines instructional design as "the process of deciding what methods of instruction are best for bringing about desired changes in student knowledge and skills for a specific course content and a specific student population" (p. 7). Some of the instructional design models or theories emphasize the micro-level of instructional strategies (e.g. Component Display Theory, Merrill, 1983). Some other models or theories address the macro-level of instructional strategies (e.g. Elaboration Theory, Reigeluth & Stein, 1983).

Romiszowski (1981) proposes that there are four levels of instructional design which always are a part of course development: (a) the course system, (b) the "lesson" level, (c) the "instructional event" level, and (d) the "learning step" level. The degree of detail increases with the level, and a lower level can be viewed as a "black box" within the higher level of the design.

> At level 1 we will treat the course more or less as a black box, concentrating on the specification of the outputs, inputs, and control mechanisms for the course as a whole. At level 2, the course structure will become apparent in terms of the sequence and interrelationship of the lessons that will make up the course.... At level 3, the structure of each lesson becomes apparent in its overall detail. We will have a specification of the sequence of instructional events which should take place.
> ...
> Finally, at level 4, the instructional events themselves become "transparent", allowing us to see exactly how they will take place. (Romiszowski, 1981, p. 270)

This system of instructional design suggests that course development can be performed in a top-down fashion. The information collected at an upper level will propagate to a lower level, and the decisions made at the upper level might restrict the design at a lower level. An analysis of instructional design applied to courseware development as practiced by Dr. M. David Merrill (Human Technology, Inc., 1986) revealed that it was generally consistent with the four-level design approach. The first three levels of design, (course, lesson, and instructional events), correspond to the identification, organization, and design of modular instructional units

(Merrill, 1987). The levels above are all concerned with macro-level design strategies. However, level four (learning step design), corresponds to a micro-level design strategy. It involves the design of transaction shells, which are described in detail in Merrill & Li (1988). The assumption is that the output of the macro-level design is a prescription for which transactions should be used, and how a transaction should be set up to teach a subject.

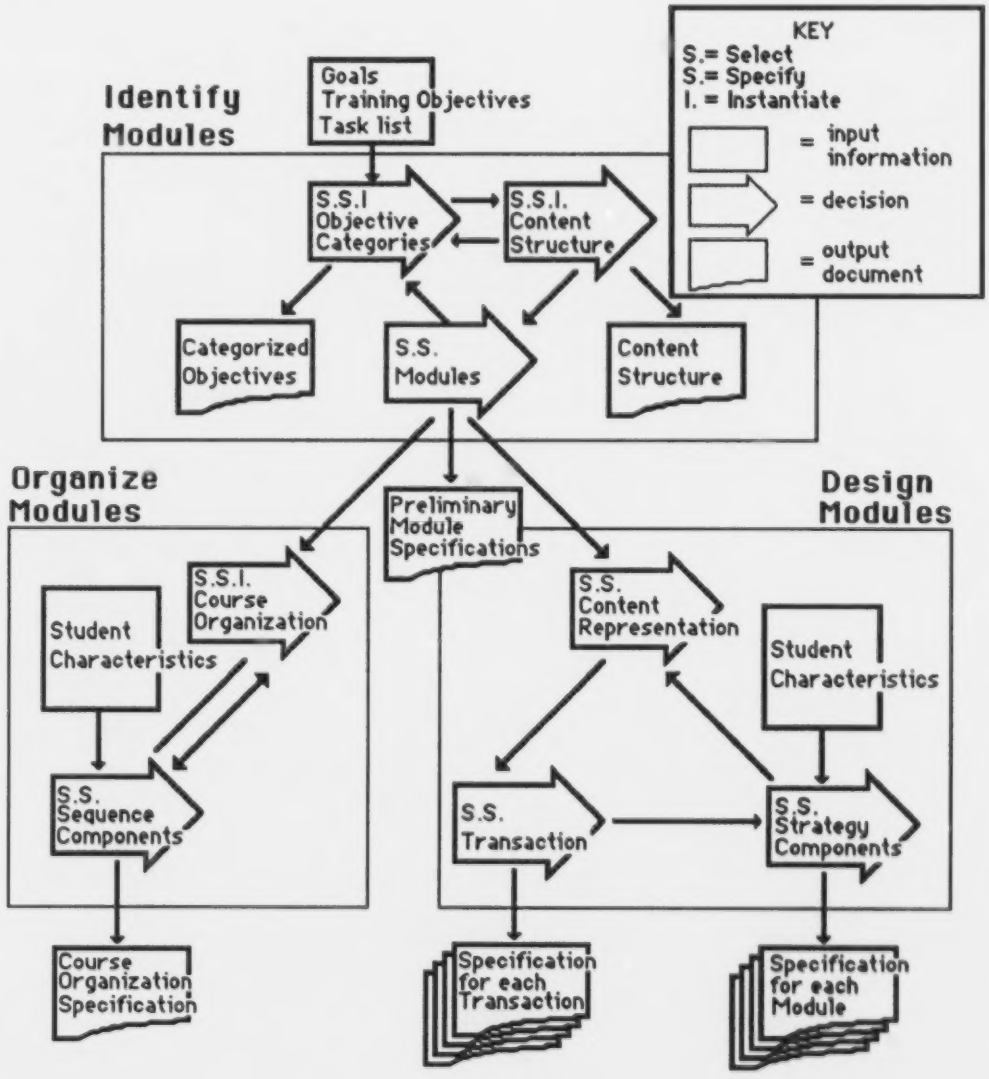


Figure 2.2. The Instructional Design Model Simulated by ID Expert.

Figure 2.2 illustrates the Instructional Design Model (IDM) in terms of the required input/output of each design level. Rectangle boxes indicate the input necessary for decisions. Arrow boxes indicate design activities. Document boxes indicate the specification output from those design activities.

In the instructional design model, the fundamental instructional design activities involved in preparing a specification for a course production require the following eight major instructional design decisions:

- Identify each module to be included -
  - (a) Select, specify, and instantiate categorized objectives.
  - (b) Select, specify, and instantiate content structure.
  - (c) Select, and specify course modules.
- Organize, arrange, and sequence modules -
  - (d) Select and specify course organization.
  - (e) Select and specify sequence components
- Design and describe each module -
  - (f) Select and specify each module's content representation.
  - (g) Select and specify each module's transaction set.
  - (h) Select and specify each module's strategy components.

The above eight decision-makings require instructional designers to engage in three distinct activities: (a) selecting, (b) specifying, and (c) instantiating (Merrill, 1987). The selection activity refers to choosing from a taxonomy of possible design options. One sign of expertise in instructional design is indicated by the use of well established taxonomies in instructional design problem solving. It increases the efficiency of problem solving by decreasing the search spaces for optimal methods instruction on the given content and student population. For example, in Component Display Theory a single idea (content) can be classified as a fact, concept, procedure, or principle (Merrill, 1983). Upon selecting an appropriate category for the single idea, a designer limits the search of instructional methods to those that are applicable only to that category.

In many cases the decision making in category selection involves an aggregation process. In this process an expert gathers initial data and

applies judgmental knowledge to make the decision regarding which category is appropriate for the given situation. The activity of providing the initial data is called specification.

Another activity that an instructional designer expert routinely performs is instantiation. For example, when a content structure suitable for a given instructional goal is selected, the structure should be filled in with subject matter content such as a concept name, or the steps of a procedure, etc.

3. Knowledge Representation

This section presents the results of conceptualization and formalization phases of the ID Expert development. The knowledge representation strategies developed in Artificial Intelligence research are described first, and then followed by detail descriptions of the representation of the instructional design model simulated by ID Expert.

3.1. Knowledge Representation Strategies

The goal of ID Expert is to produce a design specification for an instructional system, through the application of instructional design expertise coded into the system. The design specification reflects the ways to organize the information gathered during the instructional design consultation. It is the "architect's blueprint" for what the instructional system should be like. This prototype expert system adopts a combination of knowledge representation strategies such as Frames, Rules, and Object-Attribute-Value triplets to represent various instructional design knowledge.

A frame is defined as a data structure that includes declarative and procedural knowledge in pre-defined internal relations (Barr & Feigenbaum, 1981). The frame describes a class of objects. It consists of a collection of "slots", that describe properties of the objects. Rich (1983) describes the situation where frame based reasoning is in evidence:

> There exists a great deal of evidence that people do not analyze new situations from scratch and then build new knowledge structures to describe those situations. Instead, they have available in memory a large collection of structures representing their previous experience with objects, locations,

> situations, and people. To analyze a new experience, they evoke appropriate stored structures and then fill them in with the details of the current event. (p. 229)

The type of reasoning that Rich describes is seen in most of the existing instructional design theories, and models. For example, Gagne and Brigg's prescriptive model of instruction can be viewed as filling in subject matter details into the "slots" of a prerequisite objective hierarchy (Aronson & Briggs, 1983). Reigeluth and Merrill's Elaboration Theory of Instruction proposes that there are four major types of relationships that are important for purposes of instruction: conceptual, procedural, theoretical, and learning-prerequisite. Knowledge structures are assumed to be based on these relationships. Filling in the knowledge structure with subject matter is a crucial part of the instructional design activities (Reigeluth & Stein, 1983).

In a frame-based reasoning system, frames constitutes the declarative knowledge structures that organize the representation of static facts of a given domain. The primary reasoning process often consists of selecting a frame suitable for the current situation, and filling in the details called for by its slots. A slot may be filled in with default value(s) determined by an expert's prior experience with the situation that the frame describes. When specific information for the slot is not available the expert may continue the reasoning process by applying the default information. However, in most cases the information for a slot must be derived heuristically. also another important aspect of a frame based reasoning system is "procedural attachment, where procedural knowledge is associated with a particular slot (Rich, 1983, p. 230). An attachment is a sequence of instructions for deriving a value of the slot. the instructional may combine information from other slots of the frame, or from other frames (Harmon & King, 1985). If the derivation of information for a slot is considered as a sub-goal fo the system, the procedural attachment drives the reasoning or problem-solving behavior of the system to reach the goal.

The procedural aspect of knowledge representation is algorithm based. "A procedural representation of a fact [information] is a set fo instructions that, when carried out, arrive at a result consistent with the

fact [information]" (Harmon & King, 1985, p. 44). The set of instructions is referred as a plan for the use of the information represented by the procedure (Rich, 1983). A large portion of instructional design knowledge is procedural and plan based. Instructional design models often suggest a series of steps to be carried out in the design process (Reigeluth, 1983). They can be viewed as generic algorithms in carrying out design activities. Planning knowledge applied to such design activities as decomposing and sequencing the course into modules, and the modules into transactions (Human Technology, Inc., 1986). This kind of procedural and planning knowledge is suitable for an algorithmatic-based representation.

Not all knowledge needed for solving a problem can be represented by structure-oriented frames alone. An expert frequently uses guesses, rules of thumb, or heuristics when solving problems. This type of knowledge is called judgmental knowledge. Rules are used to represent the judgmental knowledge, and are expressed in IF-THEN statements, where the IF-part is the premise, and the THEN-part is the conclusion. When a rule is invoked to derive a conclusion, the system checks the premise of the rule to see whether all the clauses in the premise are true. If so the conclusion is said to be true. If any of the clauses in the premise are false the system stops examining the rule. Each clause can be understood as a sub-goal for reaching the goal represented by the conclusion of the rule.

Both procedural and declarative representation are powerful strategies. Although a piece of information can be represented using either of the strategies, the procedural representation focuses on the dynamic aspect of the information, and the declarative representation focuses on the static aspect of the information. In comparison, procedural representations are more efficient, but harder to maintain, where as declarative representations are more easier to be understood (Harmon & King, 1985). In ID Expert both approaches are used to represent instructional design knowledge. The efficiency of procedural representation is desired in sequencing, organizing, and planning activities. While in decision-making, the declarative representation makes the system transparent to users so that the reasoning process can be easily explained.

Another strategy used to represent instructional design knowledge is attribute-object-value triplets, denoted as A-O-V. Objects may be either physical or conceptual entities of a domain. In ID Expert, student, module, and transaction, etc. are all considered as objects. Attributes, like slots in a frame, represent general characteristics or properties associated with objects. Thus, the declarative aspect of a frame is represented by a set of attributes, defined on a set of objects with similar properties, denoted as A^n-O.

A domain expert often expects a set of possible values for the attribute in the reasoning process, based on prior experience. A-O-V triplets add a value dimension (V) to the frame representation, indicating the set of possible values that the attribute may take. For example, student motivation is a variable useful in designing instruction. It is classified into three categories: high, moderate, and low for use in ID Expert. Thus, the motivation property of students can be represented as {motivation, students, [high, moderate, low]}, where attribute (a) is the motivation, object (O) is the students, and value (V) is a set of three values: high, moderate, and low.

In many cases, when determining a value for an attribute, an expert's confidence in that value may not be definite due to lack of information during the reasoning process. For example, the motivation attribute of the students is a continuous variable. Its value can determined by a test instrument. Although one can be confident to classify those students who are on high end or low end of the percentile as high motivated or low motivated, it is difficult to classify those students who scored near the cut-points of either high/moderate or moderate/low motivation. In another words, our confidence of correct classification decreases near the cut-points, because the chance of misclassification increases. The degree of confidence on an attribute value can affect the expert's decision making. To compensate for errors, the A-O-V triplets can be modified by a number called a certainty factor (cf). The number can range from -1, (indicating definite evidence that a piece of information is false), to +1, (indicating that a piece of information is definitely true (Harmon & King, 1985). The certainty factor is a concept developed in an early expert system called MYCIN to deal with reasoning about uncertain knowledge. They are informal measures of confidence for a piece of

information in a reasoning system (Harmon & King, 1985). Each value of an attribute may be associated with a unique certainty factor denoted as V/cf.

An attribute may have multiple values denoted $(V/cf)^m$, where m indicates the number of attribute values. For example, in considering the motivation attribute for a group of students, a single value of either high, moderate, or low may not actually represent the group. One may say that the group is high motivated with a certainty factor of 0.45, and the group is moderate motivated with a certainty factor of 0.60.

Most useful systems must exploit not one but many frames. Related frames can be grouped together to form a frame system (Rich, 1983). In a frame system, some attributes are defined as a "shared slot" on two or more frames. One can view that these attributes are defined on multiple objects. The notation for "n" objects is O^n. Then a complete formal notation of the attribute-object-value triplet is $A\text{-}O^n\text{-}(V/cf)^m$, where A stands for an attribute, On means the attribute can be defined on more than one object, and $(V/cf)^m$ means multiple values can be held by the attribute. The values are modified by a belief in their certainty (Teknowledge, Inc., 1986).

A relationship is an attribute defined on two frames with a single value of either TRUE or FALSE, denoted as A-O2-V. The relationships in ID Expert connect two objects based on the semantic context. The relationship exists between two objects if its value is TRUE, and the relationship does not exist if the value is FALSE. For example a relationship PARENT-OF(P1, P2) is said to exist if P1 is the father or mother of P2. The relationship is undermined if neither TRUE or FALSE are the value for the relationship. Relationships may be categorized as one-to-one, and one-to-many (may also be null).

The one-to-one relationship defines a correspondence between a unique pair of objects. The one-to-many relationship defines a correspondence between one object and a set of objects (Horowitz & Williamson, 1984).

3.2. Mapping Instructional Design Model to Frame System

The instructional design model implemented in ID Expert contains four key concepts: (a) module, (b) content representation, (c) course organization, and (d) transaction. They are treated as primary objects on

which the system will operates. The fundamental unit of the system is the module, with which other objects are associated. Figure 3.1 shows the primary objects of the system and the relationships among them.

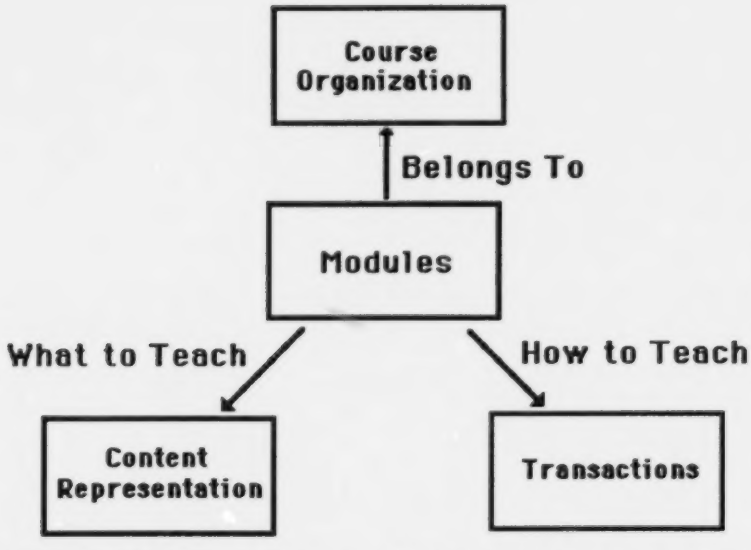


Figure 3.1. Major Objects and Their Relationships in the Instructional Design Model.

The BELONGS-TO relationship is a one-to-one relationship between the module and course organization. This relationship defines a subset of modules in the system. For a given instance of the course organization, the modules that satisfy the BELONGS-TO relationship with the instance form a subset of all the modules in the system.

The WHAT-TO-TEACHER relationship is a one-to-many relationship between content representation and module. The relationship defines the instructional content of a module. The content representation of a course varies depending on the nature of subject matter and the goals of instruction. An instance of a content representation could be a concept in a taxonomy structure or a step in a procedural hierarchy. The one-to-many relationsh indicates that more than one instance (e.g. concept or step) may be the instructional content of the same module. For example in most cases, coordinate concepts should be taught in the same module. the relationship WHAT-TO-TEACH links all the coordinate concepts to one single module.

The HOW-TO-TEACH relationship is also a one-to-many relationship between themodule and transaction. A module may serve more than one instructional functions. Each function requires a set of transactions sequenced together to present instructional materials and interact with students. The HOW-TO-TEACHER relates the instructional strategies (a collection of transactions) to a module.

Module	
Slots:	**Entries:**
Type	(depend on content structure)
Scope	((instructional , (integretive)
Function	[Overview, New Presentation, ...]
What to Teach	)) if needed, **Identify Modules**
Belongs to	If needed, **Organize Modules**
How to Teach	If needed, **Design Modules**

Figure 3.2. Module Frame

Figure 3.2 shows a frame representation for module. The three relationships, WHAT-TO-TEACH, BELONGS-TO, and HOW-TO-TEACH, are defined as slots in the module frame. Each of them has a procedural attachment that are (a) IDENTIFY-MODULES, (b) ORGANIZE-MODULES, and (c) DESIGN-MODULES, respectively. Referring to the instructional design model of ID Expert, the procedural attachments reflect the three levels of instructional design.

Procedure IDENTIFY-MODULES fills in the WHAT-TO-TEACH slot of the module frame. It first determines what kind of content structure is applicable to the given subject matter. Then, it guides the user to instantiate the content structure, and collects related information along the way. Next, it traverses through the content structure to identify the types of modules required to teach the subject, and create the instances for the module. During the traversing, ti establishes the WHAT-TO-TEACH relationship between the module frame and the frames used to build content structure of the domain.

Procedure ORGANIZE-MODULES fills in the BELONGS-TO slot of the module frame. It creates the organization for the course, and sequences all

modules and other course organization elements in an optimal instructional order.

Procedure DESIGN-MODULES fills in the HOW-TO-TEACH slot of th emodule frame. For each module in the module network, it first determines the instructional function of the module. Then, for each module function, it determines a set of transactions, required to implement the function. Finally the transactions themselves are designed in detail.

The TYPE slot is determined by the instructional content of the module, and will be discussed below in the Content Structure Representation section.

As an instructional unit in the final delivery system, a module serves one or more instructional functions. In the module frame, the function slot holds the information about module functions. The functions that ID Expert can recommend include OVERVIEW, NEW PRESENTATION, REMEDIATION, EARLY PRACTICE, LATE PRACTICE, REVIEW, and ASSESSMENT/DIAGONOSIS. Among them EARLY PRACTICE is used to help students acquire the material being taught with significant amount of learner guidance and support, while LATE PRACTICE is used to help the students increase their accuracy and peformance level on the skills being taught. Generally, LATE PRACTICE contains little or no learner guidance.

Modules are classified as either instructional or integrative category based on their functions as shown in Figure 3.3. The SCOPE slot of the module frame holds information whether a module is instructional or integrative. The inclusion of integrative modules can be justified from existing reasearches in instructional design theory development. For example, Elaboration Theory proposes summarizer and sysnthesizer as two of the strategy components of the theory (Reigeluth & Stein, 1983). both of them serve the function of integrative modules. Van Patten, Chao, & Reigeluth (1986) reviewed various strategies for synthesizing instruction.

Module Functions	
Integrative	Instructional
Overview Review Assessment/ Diagnosis	New Presentation Remediation Early Practice Late Practice Assessment/ Diagnosis

Figure 3.3. Categories of Module Function.

Content representation, course organization, transaction, and module consistute an overall paradigm for declarative knowledge representation for instructional design. Each of them belongs to one of the three phases of the instructional design model simulated in the system. The next three sections describe the knowledge representation of content structure, course organization, and transaction in detail.

3.3. Content Structure Representation

Content representation is also called knowledge representation. The knowledge structure of a given subject matter is often complex, and from an instructional design perspective, it can be classified into three content strucutres (a) concepts, (b) procedures, and (c) principles. Reigeluth and Stein (1983) suggest that any one of these may be chosen "as the most important type for achieving the general goals of the course" (p. 344). They refer to the primary content structure as the orienting structure. Knowledge from the other types can then be brought in to support the orienting structure at various levels during an elaboration process. the ID Expert handles two primary content structures: concept taxonomy, and path algorithm.

The concept taxonomy organizes the subject matter based on a conceptual framework. The procedural hierarchy shows the relations among the step of a procedure. The following sections describe the representation mechnism for concept taxonomy and path algorithm structure.

Representing concept Taxonomy

Merrill & Tennyson (1977) define a concept taxonomy as "a diagram which is constructed to indicate the subordinate, superordinate, and coordinate relationships among a set of related concepts" (p. 143). Figure 3.4 illustrates the concept taxonomy in the context of the measures of central tendency.

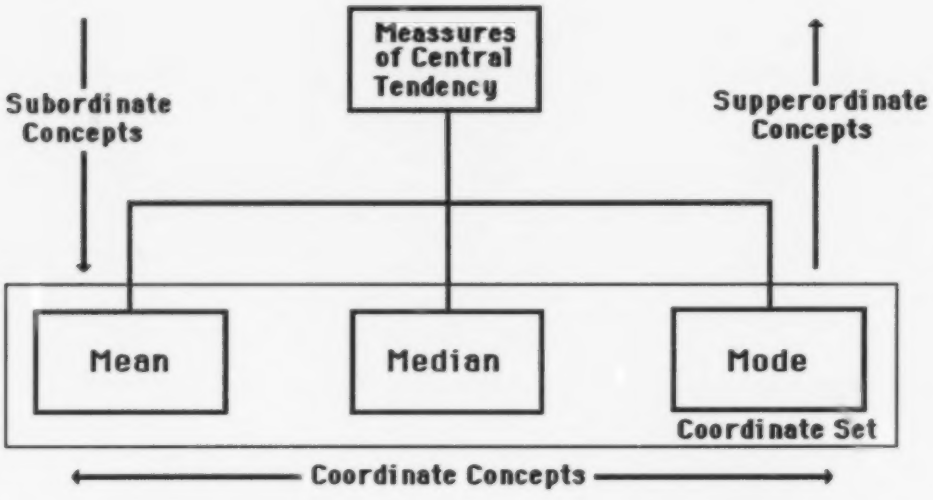


Figure 3.4. A Concept Taxonomy for measures of Central Tendency.

When a superordinate concept is divided into subordinate concepts, the subordinate concepts for a single superordinate concept form a coordinate set. The concepts in the coordinate set are called coordinate concepts. In Figure 3.4, mean, median, and mode are coordinate concepts forming the coordinate set.

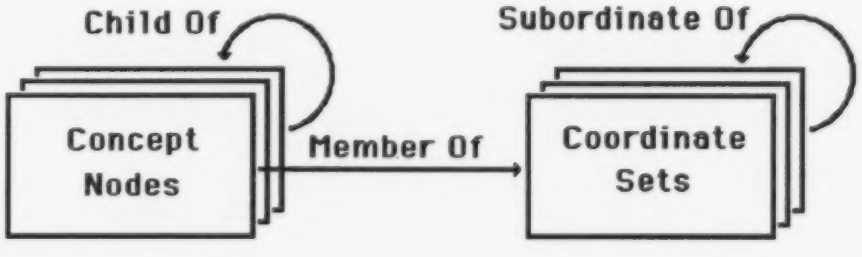


Figure 3.5. Representation of a Concept Taxonomy.

The coordinate set plays an important role in instructional design decision making. Representing a concept taxonomy requires not only to represent the relationships among concepts but also the relationships among the coordinate sets in the taxonomy. Figure 3.5 shows the representation of concept taxonomy structure, where concepts are represented as instances of a CONCEPT-NODE frame, and coordinate sets are represented as instances of a COORDINATE-SET frame. The figure shows that the instances of the CONCEPT-NODE and the COORDINATE-SET frames are connected by three relationships: (a) CHILD-OF, (b) SUBORDINATE-OF, and (c) MEMBER-OF.

The CHILD-OF relationship is a one-to-many relationship among instances of CONCEPT-NODE frame. The instances of the CONCEPT-NODE and the CHILD-OF relationships among them establish a tree structure representation of the concept taxonomy. The subordinate, superordinate, and coordinate relationships of concepts defined in terms of the CHILD-OF relationship are as follows:

Definition 3.1

Let S1 and S2 be the two instances of a CONCEPT-NODE frame. If CHILD-OF(S1, S2) is true, then S1 is said to be the subordinate concept of S2, and S2 is the superordinate concept of S1.

Definition 3.2

Let S1 be an instance of a CONCEPT-NODE frame. The coordinate set of S1 consists of those Ss, (S is an instance of CONCEPT-NODE), such that CHILD-OF(S, S1) is true.

The SUBORDINATE-OF relationship is a one-to-many relationship among COORDINATE-SET frames, and organizes these frames into an internal tree structure. Figure 3.6 illustrates a COORDINATE-SET tree generated from its corresponding concept tree.

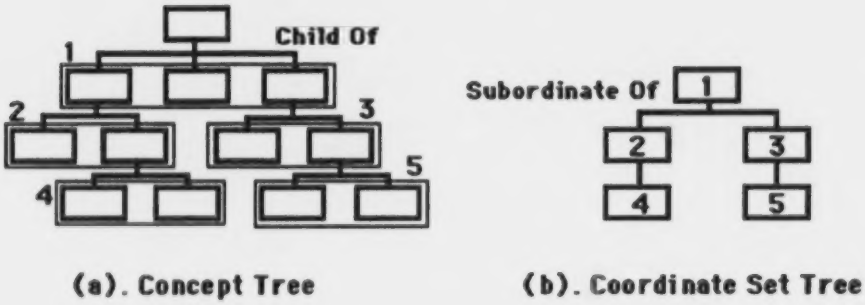


Figure 3.6. A Concept Tree and Its Corresponding Coordinate Set Tree.

The MEMBER-OF relationship is also a one-to-many relationship which establishes the correspondence between the instances of a CONCEPT-NODE frame, and a COORDINATE-SET frame.

Concept Node	
Slots:	**Entries:**
Embedded	[Embedded, Isolated]
Abstract	[Abstract, Concrete]
Attribute Number	[Many, Few]
Representation	[Text, Illustration]
Child Of	**Analyze Concept Taxonomy**
Member Of	**Analyze Concept Taxonomy**

(a) CONCEPT-NODE frame.

Coordinate Set	
Slots:	**Entries:**
Relationship	[Coordinate, Independent, ...]
Focus	[Discriminate, Identify,...]
Subordinate Of	**Analyze Concept Taxonomy**
Member Of	**Analyze Concept Taxonomy**

(b) COORDINATE-SET frame.

Figure 3.7. Frames for Concept Taxonomy.

Figure 3.7 shows a declarative representation of a concept taxonomy with the key slots of the CONCEPT-NODE and COORDINATE-SET frames. One of the characteristics in the representation of knowledge structure is that "a representation reflects the method of analysis used to produce it" (Donald, 1987, p. 190). Donald additionally states that

> One must analyze a schema [knowledge structure] into its component parts, relationships, and organizing structure in order to represent it. The representation of subject matter should therefore suggest the analytic process necessary to achieve it. (p. 190)

The analytic process that generates the concept taxonomy representation, is considered a procedural attachment to the three relational slots defined in the CONTENT-NODE and COORDINATE-SET frames. The procedure expands the concept and coordinate tree recursively, in a depth-first order. During the process of building a taxonomy tree, the system collects information about the properties of the concepts. It directs the reasoning process in determining the concept relationship, and learning focus of each coordinate set generated. There are four properties of a concept providing useful information in the design of the course.

First, a concept can be either ABSTRACT or CONCRETE. An ABSTRACT concept is difficult to show directly and must usually be inferred from events or situations. Several concepts used here such as module, transaction, and course organization are all ABSTRACT concepts. A CONCRETE concept can be directly observed, for example a dog, a cat, or a sail boat.

Second, a concept may be EMBEDDED or ISOLATED. An EMBEDDED concept refers to the concept presented together with the surrounding context. An ISOLATED concept does not require surrounding context in its presentation.

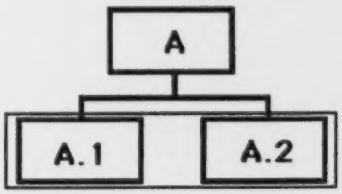
Third, the number of attributes a concept has influences learning difficulty. A concept with MANY attributes is usually harder to learn than the one with FEW attributes.

Finally, the presentation form of a concept is important. Generally speaking a concept represented by some GRAPHIC presentation such as a

chart, a drawing, or a picture is more easily understood than the one with TEXT presentation only.

Donald (1983) found that the use of tree structure worked best with well-structured knowledge because the relationships between concepts tended to be hierarchical, leading to tree structures with many tight links. In a less well-structured domain, the methods such as graphic analysis, and protocol analysis are commonly used (Donald, 1987). In attempting to use a unified structure to represent both well-structured and less well-structured knowledge, we introduced a notion called coordinate concept relationships to supplement the tree representation of a concept taxonomy.

Five coordinate concept relationships have been identified as INDEPENDENT, COORDINATE, ORDERED, OVERLAP, and CONTINUUM. An INDEPENDENT relationship means that the instances of one category in the coordinate set are very dissimilar to instances in a neighboring category such as the concepts of dog and fish. INDEPENDENT concepts do not require learner to classify or discriminate among them. A COORDINATE relationship illustrates that the instances of one category in the coordinate set are similar to instances in neighboring category. An ORDERED relationship refers to the instances of the various categories in the coordinate set, and can be ordered on some dimension. An OVERLAP relation means that boundaries of the various categories in the coordinate set are unclear. Instances close to these boundaries are difficult or impossible to clearly identify as belonging to one category or the other. A CONTINUUM relationship refers to one where there are ill-defined or no boundaries between various categories. In well-structured knowledge, INDEPENDENT and COORDINATE relations dominate, while in less well-structured knowledge domain, OVERLAP, and CONTINUUM relationships dominate.



A.1 and A.2 are coordinate concepts of A.

IDENTIFY:
Student will be able to identify instances of A.

DISCRIMINATE:
Student will be able to discriminate instances of A.1 and A.2 from one another.

RANK:
Student will be able to rank/order instances of A on the dimension C. (C is the dimension along which the concepts are ordered).

KINDS OF
Student will be able to identify instances of A as instances of A.1 or A.2.

GENERALIZE:
Student will be able to identify instances of A.1 and A.2 as instances of A.

Figure 3.8. Coordinate Set Focuses and Their Associated Instructional Objective Patterns.

Since the Behavioral-Objective-Movement, taxonomies have been proposed to classify instructional objectives. ID Expert classifies six types primary focus for learning the concepts in a coordinate set. They are (a) IDENTIFY, (b) DISCRIMINATE, (c) GENERALIZE, (d) RANK, (e) KINDS-OF, and (f) PLACE-HOLDER, represented by the FOCUS slot of the COORDINATE-SET frame. Each focus is associated with an instructional objective pattern as shown in Figure 3.8. The PLACE-HOLDER focus provides the consistency and coherence of the system, though it does not have any instructional meanings. Once the learning focus is identified content sensitive instructional objectives are generated by filling the objective patterns with the name of concepts in the coordinate set.

In reality the relationships among the concepts in a coordinate set may lead to restrictions on learning activities necessary for learning the

concepts. Figure 3.9 shows the dependency between COORDINATE-SET-FOCUS and COORDINATE-SET-RELATIONSHIP. The "N/A" square means that the COORDINATE-SET-FOCUS is not applicable for the corresponding COORDINATE-SET-RELATIONSHIP. For example, it is not meaningful to teach students to discriminate among concepts that OVERLAP or form a CONTINUUM. During the reasoning process if the system is not able to get the information on COORDINATE-SET-FOCUS, default values (squares marked with "DFT." in figure 3.9) are used so the reasoning process will not be interrupted. It is assumed that the best way to teach concepts with INDEPENDENT relationships is IDENTIFICATION; the best way to teach concepts with COORDINATE relationships is DISCRIMINATION; and the best way to teach concepts with ORDER, OVERLAP, and CONTINUUM relationships is RANKING or ORDERING.

Relation \ Focus	Discriminate	Rank/Order	Identify	Generalizatio	Kinds Of
Independent		N/A	DFT.		
Coordinate	DFT.	N/A			
Order		DFT.			
Overlap	N/A	DFT.	N/A		
Continuum	N/A	DFT.	N/A		

Figure 3.9. Dependency Between Coordinate Set Relationship and Coordinate Set focus.

Modules in ID Expert are classified into instructional and integrative categories. There are four types of instructional modules in a concept taxonomy: (a) SINGLE-NODE, (b) COORDINATE, (c) SUPERORDINATE, and (d) COORDINATE-SUPERORDINATE as shown in Figure 3.10.

Rule-1: If Coordinate Relationship is Independent and
Coordinate Focus is Identify
Then Module Type = Single Node.

Rule-2: If Coordinate Relationship is Coordinate and
Coordinate Focus is Discriminate
Then Module Type = Coordinate.

Rule-3: If Coordinate Relationship is Independent and
Coordinate Focus is Generalize
Then Module Type = Superordinate.

Rule-4: If Coordinate Relationship is Coordinate and
Coordinate Focus is Generalize
Then Module Type = Coordinate Superordinate.

Figure 3.10. Module Types and Sample Rules for Their Selection.

A SINGLE-NODE type always contains a single concept. A COORDINATE type consists of all concepts in a coordinate set. A SUPERORDINATE type includes only one concept and its superordinate concept. A COORDINATE-SUPERORDINATE type comprises all concepts in a coordinate set and their superordinate concept. The selection of module type required for a given set of concepts depends on the COORDINATE-SET-RELATIONSHIP, and COORDINATE-SET-FOCUS information. Figure 3.10 also gives some sample rules to represent judgmental knowledge used in determining module types.

Many cases in actual instructional design situation, more than one coordinate set focus may be selected for a given coordinate set. For example, an instructional designer may decided that both classification and discrimination tasks may be necessary for students to learn the concepts in the same coordinate set. However, selecting too many coordinate set focus may also confuse the students. To provide flexibility of multiple coordinate set focus, but also to control the number of focus selected low, ID Expert allows two coordinate set focuses to be selected for a coordinate

set. Then, conflicts arise when more than one module type are recommended for the same coordinate set. Continuing with our example, since classification focus requires COORDINATE module type, and discrimination requires COORDINATE-SUPERORDINATE module type, decision must be made regarding which module type to chose when conflicts happen. Figure 3.11 shows a set of conflict resolution rules to solve the problem.

Single Node + Coordinate = Coordinate

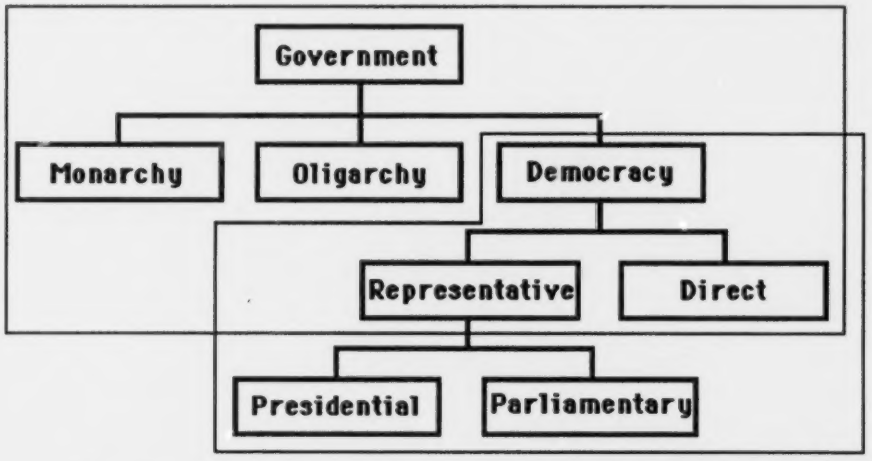
Single Node + Superordinate = Superordinate

Coordinate + Superordinate = Coordinate Superordinate

Any types + Coordinate Superordinate = Coordinate Superordinate

Figure 3.11. Conflict Resolution Rules for Module Type Determination.

The rules are specified in the form of A+B=C, which means that if module type A and B are both required for a given coordinate set, then recommend module type C instead because C contains both A and B. For example, COORDINATE and SUPERORDINATE types contain the SINGLE-NODE type, and COORDINATE-SUPERORDINATE contains all other module types.



Integrative modules are required to teach concepts: GOVERNMENT, and DEMOCRACY.

Figure 3.12. Integrative Modules for a Concept Taxonomy.
(The taxonomy adopted from Reigeluth & Stein, 1983, p. 350)

There are two types of integrative modules (preintegrative, and postintegrative), that always appear as pairs in course organization. The method used to determine whether a pair of integrative modules is required to teach a concept is quite straight forward. For a given concept node, an integrative module pair is required, if at least one of the subordinate concepts of the node is not the terminal node in the concept taxonomy. A terminal node refers to the node that contains no subordinate node in the concept taxonomy. Figure 3.12 illustrates the formation of integrative modules for a concept taxonomy, where integrative module pairs are required for GOVERNMENT, and DEMOCRACY concepts.

Representing Procedural Hierarchy

A procedural hierarchy is a structure that shows the relationships among the steps of a single procedure, through the "procedural-prerequisite relations" (Reigeluth, Merrill, & Bunderson, 1978, p. 110). The procedural hierarchy can be constructed by decomposing a task into a set of subtasks called steps, based on the procedural-prerequisite relations. Each step can then be further analyzed and decomposed until students can perform a step without now learning. Figure 3.13 shows an example of procedural hierarchy in the context of a spreadsheet application.

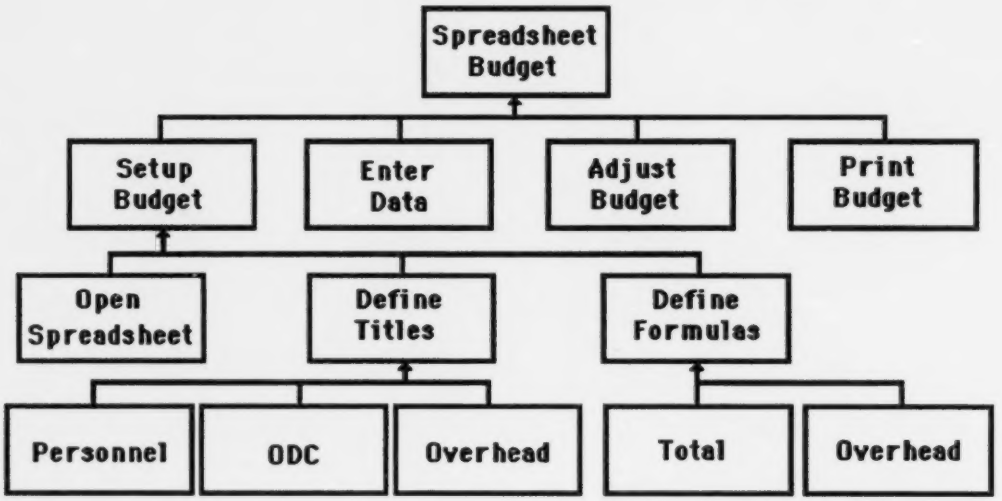


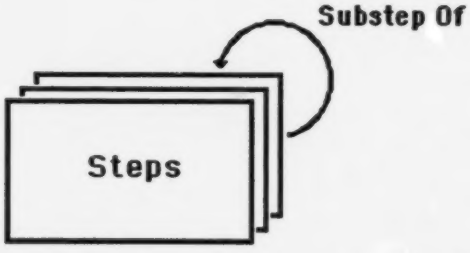
Figure 3.13. A Procedural Hierarchy for Spreadsheet Application.

In the procedural hierarchy, the arrow between two steps on different levels means that the lower step must be performed before the higher step (Reigeluth & Stein, 1983, Reigeluth, et al. 1978).

The underlying representation mechanism for procedural hierarchy is the STEP frame as shown in Figure 3.14. The STEP frame consists of four key slots: NAME, ENTRY-LEVEL, JOB-AIDS, and SUBSTEP-OF. NAME holds the label for the step. ENTRY-LEVEL slot holds information on whether the step is specified at the entry level of prospective students. The JOB-AIDS slot contains information on whether there are some kinds of job aids available for students to use while performing the step. The SUBSTEP-OF is a one-to-many relationship which link the instances of steps to an internal tree representation of the procedural hierarchy. Thus, in ID Expert, the SUBSTEP-OF indicates the procedural prerequisite relationship.

Step	
Slots:	**Entries:**
Name	Any Text
Entry Level	[Yes, No]
Job Aids	[Yes, No] Default = No.
Substep Of	If needed, **Analyze Procedure**

(a) STEP Frame for Procedural Hierarchy.



(b) SUBSTEP-OF Relationship Links Steps into a Tree Representation.

Figure 3.14. Mechanism for Representing Procedural Hierarchy.

A formal definition of the procedural-prerequisite relationship is as following:

Definition 3.3

> For two steps, X and Y in a single procedure, X and Y are said to have a procedural-prerequisite relationship if the performer must do X before he can do Y denoted by SUBSTEP-OF(X, Y).

For a given task, a procedural hierarchy can be constructed from several perspectives. A procedural hierarchy is instructionally useful if each step in the hierarchy has no more than eight (8) substeps. Formally defined as:

Definition 3.4

> For all instances of a procedural hierarchy, S_i, such that the number of S_j with SUBSTEP-OF(S_i, S_j) being true is less than or equal to seven (7).

The classification of a procedural hierarchy as instructionally useful, is a result of human information processing capacity research which indicates that most people can "reliably recall seven items" (DoBois, Alverson, & Staley, 1979, p. 110). Generally speaking, at a given step level, if the step contains more than seven (7) substeps, it is probably be identified as too specific at that level (Reigeluth & Merrill, 1984).

The SUBSTEP-OF slot is filled by procedure attachment ANALYZE-PROCEDURE. The procedure is a simplification of the Extended Task Analysis Procedure (ETAP) developed by Reigeluth & Merrill (1984). We only implemented the procedural analysis part of the ETAP. The procedure guides a user to build the procedural hierarchy. For each step of a procedure, decisions should be made regarding whether to further analyze the step. Figure 3.15 lists the rules which terminate the analysis procedure. In fact performing a task in many real job

Rule-1: If the current step is specified at the entry level of target student
Then no further analysis is necessary.

Rule-2: If the current step is not specified at the entry level of target student and there is a job aid for performing the task and the target students can perform the step with the help of job aid
Then no further analysis is necessary.

Figure 3.15. Termination Rules for Analyzing a Procedure.

setting usually has some sort of job aids available to the performer. The instruction will be more "real-world" oriented if the use of job aids is included in the instruction. In some cases if the job aids are not designed for the target students' entry learning level, further analysis of the job aids will be necessary for successful instruction (Reigeluth & Merrill, 1984).

In ID Expert a procedural hierarchy only requires one instructional module type, the COORDINATE type. Each instructional module consists of the steps in the subprocedure defining the module. It is assumed that the most efficient way to teach a subprocedure is to teach them as one instructional entity, which is represented by a module.

Integrative modules are also determined in pairs. The method used in analyzing a concept taxonomy applies to analyzing a procedural hierarchy. Integrative modules are required for a procedure if at least one of the steps of the procedure contains sub-procedures. As an example, both SPREADSHEET-BUDGET, and SETUP-BUDGET procedure in Figure 3.13 requires integrative modules.

3.4. Course Organization Representation

Course organization primarily deals with the sequence aspect of instructional design. That is, what to present, when, and in what order. The existing research on sequencing can be categorized as micro sequencing and macro sequencing. Micro sequencing deals with "ordering instruction for teaching individual ideas", and macro sequencing is concerned with "ordering multiple ideas in course/curriculum design" (Van Patten et al., 1986, p. 454). Several instructional design theorists suggest that there are four different levels of sequencing problem (Gagne & Briggs, 1974; Romiszowshi, 1981). Each level defines a fundamental element in a course structure. The research indicates that a conventional course structure contains four fundamental elements: course, units, lessons, and learning events as shown in figure 3.16(a).

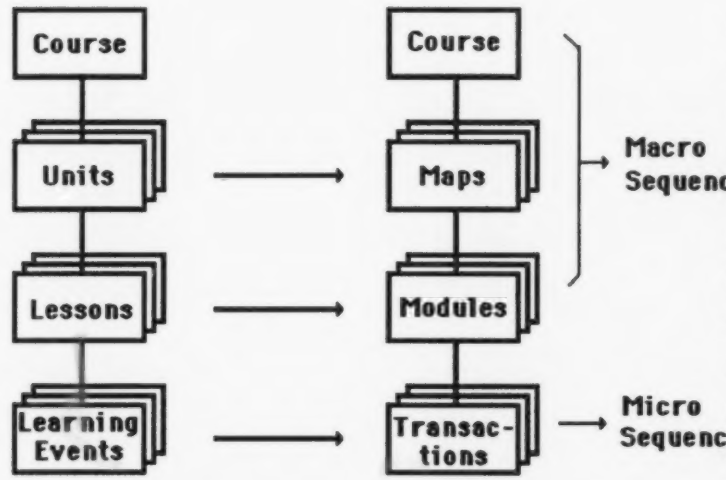


Figure 3.16. Comparisons of Course Structure in Conventional Instruction and ID Expert

Figure 3.16(b) demonstrates the relationship of the conventional course structure and the one used in ID Expert, where units correspond to INSTRUCTIONAL-MAPS; lessons correspond to MODULES; and learning events correspond to TRANSACTIONS. Both MODULE and MAP are considered as two fo the three elements of course organization. The other one is INSTRUCTIONAL-MENU. Figure 3.17 shows the definition for COURSE-ORGANIZATION-ELEMENT frame.

Course Organization Element	
Slots:	**Entries:**
Type	[Module, Menu, Map]
Consists Of	If needed, **Organize Modules**
Next Element	If needed, **Sequencing**

Figure 3.17. Course Organization Element Frame.

Figure 3.18 shows how the three types of COURSE-ORGANIZATION elements are associated with each other to form an INSTRUCTIONAL-MAP structure.

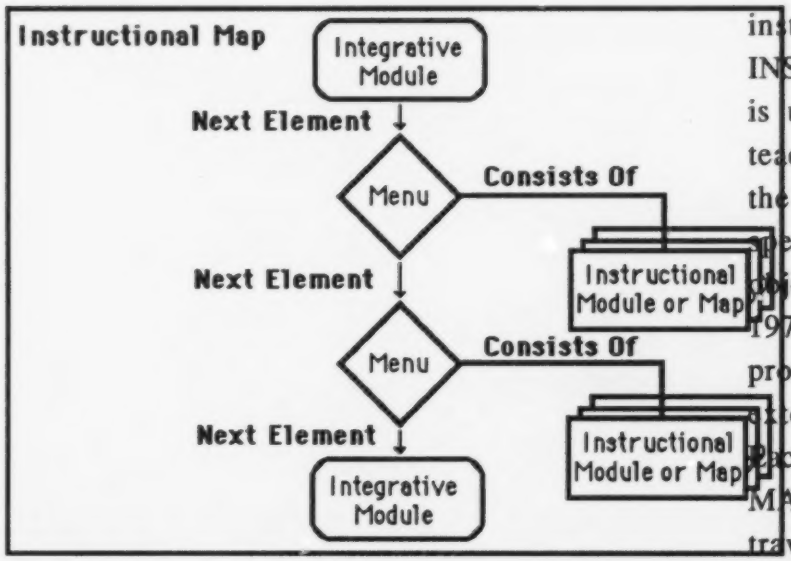


Figure 3.18. Instructional Map Representation.

In conventional instructional design, an INSTRUCTIONAL-MAP is used to show "the teaching sequence and the interactions for specific behavioral objectives" (Briggs, 1977, p. 110). The prototype ID Expert extends this notion. Each INSTRUCTIONAL-MAP also shows the traversability among a subset of the module network, once a student

takes the instruction contained in the MAP. Research shows that learning can be maximized if an instructional system can adapt itself to the needs of individual student. In consideration of this, the CONSISTS-OF relational slots applies to the COURSE-ORGANIZATION-ELEMENT of an INSTRUCTIONAL-MENU type. The slot holds the information about the static nature of the menu. It is associated with a complete set of instructional modules from that students can select during instruction. However, in the actual instruction, only a subset of the modules will be presented to the students based on a set of sequence rules in the MENU. The set of sequence rules represent the dynamic aspect of the MENU. The determine when a student can traverse between modules, and whether the system or the student should make the decision.

The NEXT-ELEMENT slot of the COURSE-ORGANIZATION-ELEMENT frame is used to sequence integrative modules, and instructional menus to form a high level instructional sequence within an instructional map as in Figure 3.18.

A COURSE-ORGANIZATION-ELEMENT of INSTRUCTIONAL-MAP type represents a subset of the module network of a course organization. The connectivity of all the subsets of the module network is done by nesting subsequent instructional maps. By associating an INSTRUCTIONAL-MAP with an INSTRUCTIONAL-MENU, the MENU takes control of when a student may traverse to other instructional maps, and whether the decision is made by the system or the student.

Reigeluth (1979) defined three continua for sequencing instruction: general to detailed, simple to complex, and abstract to concrete. In ID Expert, the general to detailed continuum is the primary sequencing principle used to decompose the concept or procedural hierarchy into course organization. According to Reigeluth and Stein (1983):

> The general-to-detailed continuum refers primarily to a continuum formed by subdividing ideas (either concepts or procedures) or by lumping ideas (subordinate concepts or subprocedures) together. General has breadth and inclusiveness (lots of things lumped together), whereas detailed is usually narrow (subdivisions). (p. 344)

Figure 3.19 shows the sequencing algorithm used in ID Expert for a concept taxonomy. Although the general sequencing principle is from general to detailed, the continuum is reversed if the coordinate set requires a COORDINATE-SUPERORDINATE module and the COORDINATE-SET-FOCUS is GENERALIZATION. In which case subordinate concept are taught first.

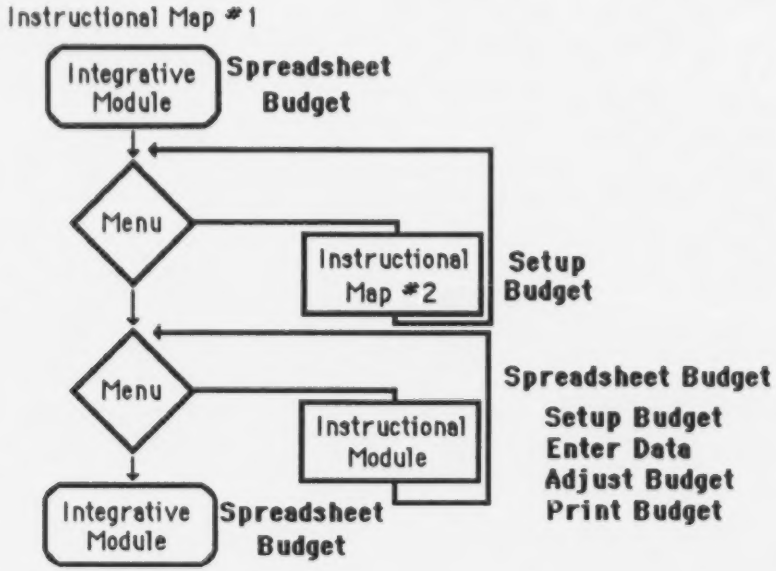
The procedure is defined recursively. By using the root node of a concept taxonomy as the first input to the algorithm, it will generate all the COURSE-ORGANIZATION-DIAGRAMS for the concept taxonomy.

For a procedural hierarchy it is assumed that when a step of the procedure is subdivided into substeps, the substeps become prerequisites of the step. To

```
Input: A Concept Node

1. Create four course orgainzation elements(COE):
   a preintegrative module for teaching the concept node;
   an instructional menu called menu1;
   an instructional menu called menu2;
   a postintegrative module for teaching the concept node.

2. If children of the concept node belong to same module
   Then the COE is the module
   Else
     If children of the concept node belong to N modules
     Then each module is the COE parallel to each other
     Else COE is a place holder.

3. If one of the children nodes belongs to a coordinate-
     superordinate module and
     coordinate set focus is generalization
   Then associate the COEs in step 2 to menu2,
        associate the COEs in step 4 to menu1
   Else associate the COEs in step 2 to menu1,
        associate the COEs in step 4 to menu2.

4. For each child of the concept node
     If all children of the child is a terminal node and
       all children of the child belong to same module
     Then the COE is the module
     Else the COE is an instructional map.

5. For each child of the concept node
     If the child is associated with an instructional map
     Then make the child as input and start from 1.
```

Figure 3.19. Sequence Procedure for Concept Taxonomy.

learn to perform the step, the students must learn how to perform all the substeps. According to Gagne and Briggs (1974), a "learning hierarchy is an arrangement of intellectual skill objectives into a pattern which shows the prerequisite relationships among them" (p. 109). If it is assumed that the terminal nodes in the procedural hierarchy are specified at the entry level of the target population, the procedural hierarchy can be viewed as equivalent to a learning hierarchy. The sequencing algorithm for a procedural hierarchy applies a simple to detailed continuum for sequencing integrative modules, and a learning hierarchy (bottom up or detailed to simple) continuum for sequencing instructional modules. For example, the sequence to teach the procedural hierarchy in Figure 3.13 would be: (a) teach the integrative module of SPREADSHEET-BUDGET; (b) teach the integrative module of SETUP-BUDGET; (c) teach the instructional modules of DEFINE-TITLES, and DEFINE-FORMULAS; (d) teach the instructional module of SETUP-BUDGET; and (e) teach the instructional module SPREADSHEET-BUDGET. Figure 3.20 shows the corresponding COURSE-ORGANIZATION diagrams.



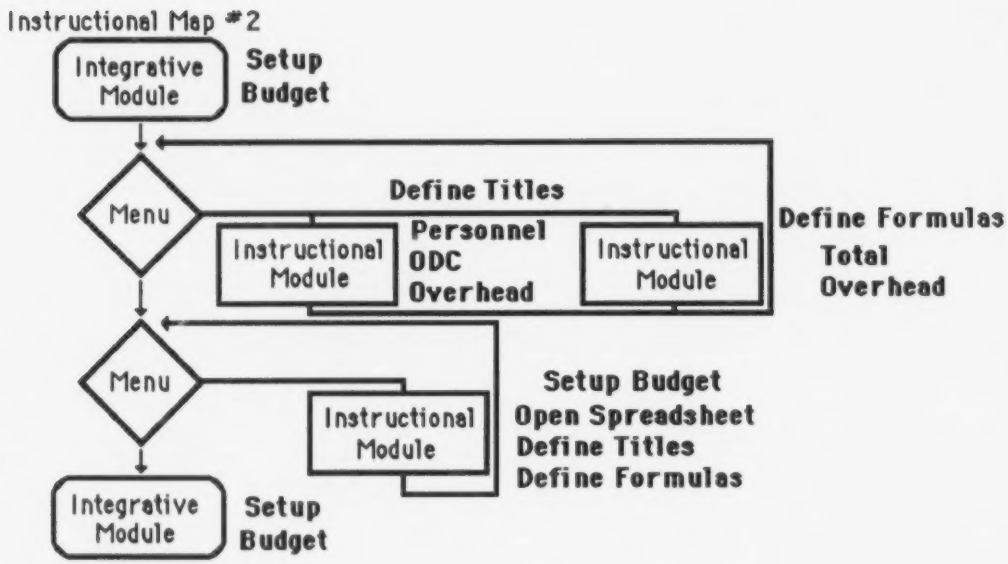


Figure 3.20. Course Organization Diagrams for Spreadsheet Budget.

3.5. Transaction Representation

Module design consists of three activities: (a) select the functions of a module; (b) select the transactions for each function; and (c) design the transactions. Since the associations between transactions and a module are established through module function, an intermediate concept (object)

called FUNCTION was introduced. The relationships among modules, functions, and transactions are illustrated in Figure 3.21.

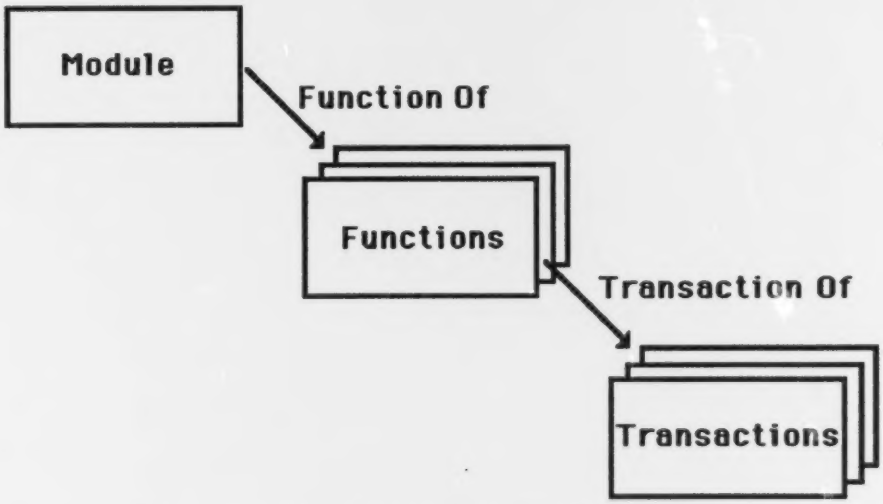


Figure 3.21. Transaction Representation.

FUNCTION-OF is a one-to-many relationship which connects the instances of FUNCTION frame to a particular instance of MODULE frame. TRANSACTION-OF is also a one-to-many relationship which connects the instances of TRANSACTION frame to a particular instance of FUNCTION frame.

4. Implementation

S.1, a general purpose expert system shell developed and marketed by Teknowledge, Inc., was chosen to implement the ID Expert, because it supports the knowledge representation required in the prototype system. The frames, inference rules, and control strategies described in the previous section (conceptualization, and formalization phases) were converted to S.1 codes. The purpose of this section is to discuss the implementation issues addressed in developing the prototype ID Expert using the S.1 expert system shell.

4.1. Stages of Implementation of Related Tasks

To solve an instructional design problem, a designer should have analytical methods to access subject domain to reveal the knowledge

structure of the domain. The designer should also be able to organize the subject knowledge to derive the final course organization. In doing so, the designer must take into consideration many factors such as the characteristics of target population, the intended performance level, etc. Because the number of design variables is so large, the instructional design task is non-trivial. Even a prototype system like ID Expert requires a large number of rules and other knowledge base objects.

Relative to conventional software development, when building an expert system, knowledge engineers and domain experts must maintain a tight interaction for the project to be successful. Figure 4.1 illustrates the interactions between a knowledge engineer and a domain expert.

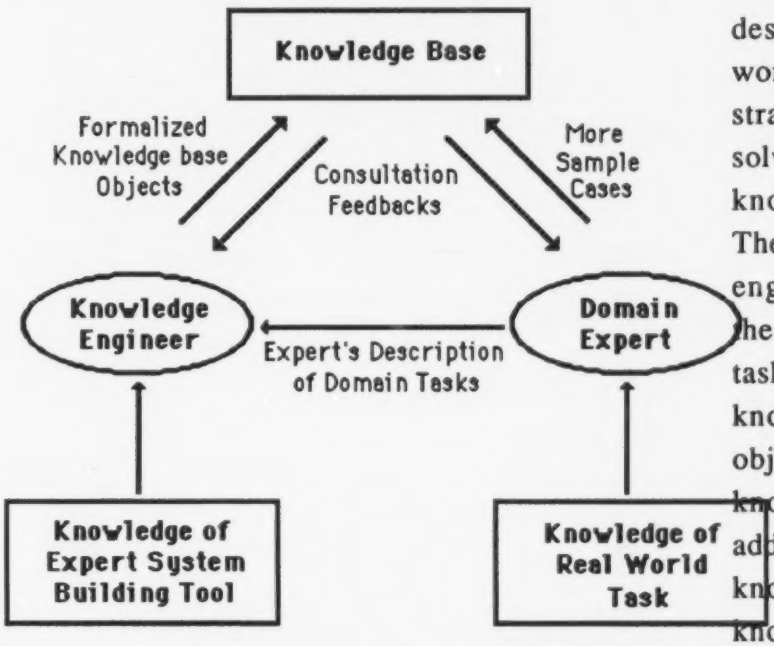


Figure 4.1. Interactions between a Domain Expert and a Knowledge Engineer at the Implementation Phase.

The expert describes the real world task and strategies in problem solving to the the knowledge engineer. The knowledge engineer formalizes the description of the task, codes the knowledge as the objects in the knowledge base, and adds the new knowledge to the knowledge base. The knowledge engineer employs sample cases to test the knowledge base to eliminate the coding errors. The expert runs sample cases through the system, compares the consultation behavior of the system with his own in solving the kinds of problems described in the sample cases, and provides

more precise description of the domain task to the knowledge engineer. The nature of knowledge acquisition determines that the implementation of the system is an incremental process.

Table 4.1. List of Additive Knowledge Bases of ID Expert.

kb_i	DESCRIPTIONS
0	Basic classes, class types, & attributes
1	Audience attributes
2	Reasoning on coordinate set relationship
3	Reasoning about coordinate set focus
4	Displaying results of focus & relationship
5	Consistency checking of coordinate set focus
6	Reasoning module types
7	Creating course organization
8	Reasoning module functions
9	Reasoning concept characteristics
10	Reasoning transaction type I
11	Reasoning transaction type II
12	Reasoning transaction sequence
13	Reasoning transaction control mode
14	Reasoning transaction content representation
15	Printing consultation report
16	Reasoning module function (modified)
17	Reasoning transaction type (modified)
18	Reasoning student motivation
19	Structure for procedural hierarchy
20	Front-end analysis I
21	Front-end analysis II
22	Identifying modules for procedural hierarchy
23	Organizing & Design modules for procedural hierarchy

The ID Expert knowledge base consists of twenty four implementation stages composed of one primary knowledge base (KB_0), and twenty three additive knowledge bases ($kb_{i,\ i\ in\ [1..23]}$). Table 4.1 lists a brief description for each of these. An additive knowledge base is one that is added to a previously existing knowledge base, to improve problem-solving behavior of the expert system. At each implementation stage J, an additive knowledge base kb_j consists of a set of new knowledge base objects which represent the expert's problem solving behaviors, and a set of modifiers of existing objects in the expert system. The knowledge

base for implementation stage J (KB_J) thus consists of KB_0 and all kb_i, where is is less than or equial to J, expressed mathematically as:

$$KB_J = KB_0 + \sum_{i=1}^{J} kb_i \quad \text{or} \quad KB_J = KB_{J-1} + kb_J \qquad (4\text{-}1)$$

Each knowledge base constructed using formula 4-1 becomes a sub-model that "generates problem-solving behaviors that the expert can evaluate and improve" (Frederick, 1986, p. 5-34). The inclusion of each additive knowledge base in the expert system at each implementation stage often improves the expert's understanding about the task doamin. For example, KB_{15} completes the whole reasoning process for the designing of concept-taxonomy instruction. A review of KB_{15} indicated that the expert's previous estimates of certainty factors in decision making were too high. This was applicable to each instructional design variable involved in determining transactions for each module function. In many situations, almost all types of transactions are stronly recommended although they should not be. More inquires were made to gain understanding of the relationships between certainty factors, and causal relationships among instructional design variables. New approaches were used to estimate certainty factors in handling the uncertain knowledge. KB_{16} and KB_{17} reflected modification of the expert's knowledge coded in the system.

Other evidence that incremental implementation improves the expert's understanding of his own problem-solving model, is reflected in the conceptual difference in implementation of two sub-systems: concept taxonomy, and procedural hierarchy. In the concept sub-system a set of transactions are recommended for each module function. How transactions should be sequenced together is not addressed in the implementation. In other words the recommendations are restricted to the static aspects of transactions. In the procedure sub-system a transaction structure, instead of a set of transactions, is recommended. Figure 4.2 shows an example of transaction structure called Structured Learner Control Basic Instruction (SLCBI). The abbreviation "LC" stands for learner control. SLCBI suggest that procedure should be first presented in an integrative module with a DEMO transaction, which overviews the procedure. Then the students have

controls over selecting from three transactions: SHOW PARTS, EXPLORE PARTS, or NAME PARTS. Once the students successful complete this part of instruction, then they have the controls over selecting from two transactions that teach the procedure: DEMO OPERATION, and EXECUTE OPERATION. The structure illustrates the dynamic aspects of transactions as well as the static aspects. Therefore, a portion of the judgmental knowledge in designing instruction is converted to meta-knowledge. This derives the operation of the transaction structure in procedural sub-system implementation.

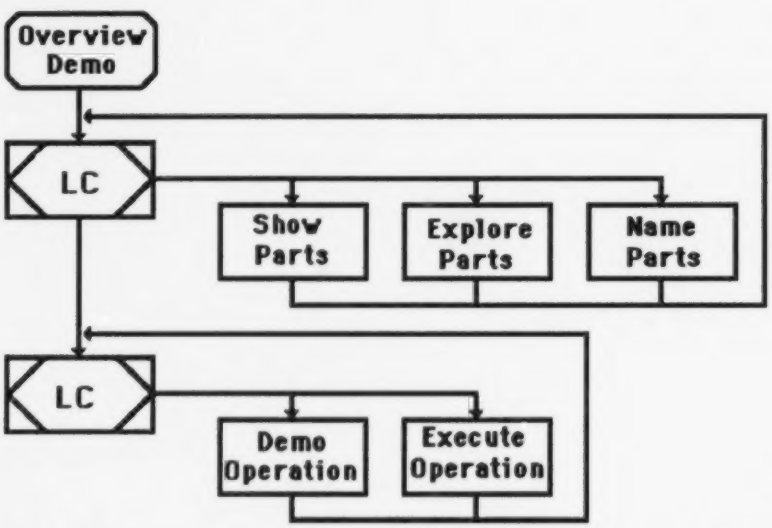


Figure 4.2. A Transaction Structure: Structured Learner Control Basic Instruction.

Table 4.2 groups the twenty three (23) additive knowledge bases into five primary tasks: (a) front-end analysis, (b) identifying modules, (c) organizing modules, (d) designing modules, and (e) procedural sub-system. Three of the five

Table 4.2. Additive Knowledge Bases Contained in Each Task Group.

Order of Implementation

TASK GROUP	ADDITIVE KNOWLEDGE BASE
Front-End Analysis	1,18,20,21
Identify Modules (concept)	2,3,4,5,6,9
Organize Modules (concept)	7
Design Modules (concept)	8,10,11,12,13,14,15,16,17
Procedural Sub-system	19,22,23

tasks (identifying, organizing, and designing modules), were devoted to the implementation of the concept taxonomy sub-system. Each is identified as a phases in the Instructional Design Model of ID Expert. The three tasks implement a majority of knowledge base since sixteen (16) of the twenty-three (23) additive knowledge bases were used.

The front-end analysis was divided into two parts. The first part simply assumes that the content structure is a concept taxonomy, and collects related audience data so taht a rapid prototype of the concept taxonomy sub-system can be constructed. The second part of the front-end analysis includes consultation on content structure determination. It integrates the two sub-systems that handle the concept taxonomy and procedural hierarchy structure, while reserving branching points for further addition of sub-systems that deal with other content structures.

Finally, the procedural sub-system task deals with the procedural hierarchy. The sub-system maintains its own declarative structure for representing the procedural hierarchy, but uses the same structure as a concept taxonomy in moudle organization and design phases.

4.2. Implementing Frame-Based System in S.1

Waterman (1985) presents a brief description of S.1:

> S.1 is a knowledge engineering language [expert system shell] for rule-based representation, but it also supports frame-based and procedure-oriented representation methods. Its principle characteristics include a backward chaining control

> scheme, built-in certainty handling mechanisms, and control blocks that support procedure-oriented representation and programming method. (p. 365)

S.1 consists of eight knowledge base objects: class, class type, attribute, value hierarchies, rule, certainty factor, control blocks, and function. All objects except value hierarchies are used to represent instructional design knowledge in ID Expert. They are categoriezed when representing the following three types of domain knowledge (a) factual, (b) judgmental, and (c) strategic knowledge (Teknowledge, Inc., 1986). The factual and judgmental knowledge are forms of declarative knowledge, and strategic knowledge, is a type of procedural knowledge.

Classes, class types, and attributes are objects to represent factual knowledge. They are essential in coding the declarative structure of a frame-based expert system. A class represents a group of objects with similar properties. To create an object is said to create an instance of a class, within which the object is defined. In Figure 4.3(a), a module class is defined as representing all modules in the system. To create a module is to create an instance of the module class. The word "any" in the NUMBER.INSTANCES field refers to a theoretically unlimited number of module instances that the system can create.

```
DEFINE CLASS          MODULE
::NUMBER.INSTANCES    any
::CLASS.TRANSLATION   "a module"
END.CLASS
```

(a). Class Definition for MODULE Frame.

```
DEFINE ATTRIBUTE      SCOPE
::DEFINED.ON          m:module
::TYPE                text
::LEGAL.VALUES        (preintegrative, postintegrative
                       instructional, place.holder)
::LEGAL.MEANS         (try.rules)
::DETERMINATION.MEANS (try.rules)
```

(b). SCOPE Slot of the MODULE Frame Defined as Attribute.

DEFINE ATTRIBUTE	MODULE.FUNCTION
::DEFINED.ON	m:module
::TYPE	text
::LEGAL.VALUES	{overview, new.presentation, old.remediation, early.practice, late.practice, review, assessment.diagnosis}
::LEGAL.MEANS	{try.rules, query.use}
::DETERMINATION.MEANS	determine.module.function

(c). FUNCTION Slot of MODULE Frame with Procedural Attachment.

Figure 4.3. Partial Implementation of the MODULE Frame Using S.1 Objects.

Attributes are used to represent properties of classes. In a frame-based representation they are defined as the "slots" of frames in the system. Thus, a complete implementation of a frame in S.1 consists of a class definition to identify the frame, and a set of attributes defined on the class as the slots of the frame. Figure 4.3 shows an implementation of the module frame with only SCOPE and FUNCTION attributes defined. The LEGAL.VALUES field of the attribute restricts the value that may be concluded for the attribute. The LEGAL.MEANS field lists the method the expert system may use to get the information for the attribute. Two common methods are to use the rules in the knowledge base to derive the value, or to directly ask the user for the information. If an attribute has a procedural attachment, a procedure (called CONTROL BLOCK in S.1) name should be listed in DETERMINATION.MEANS field as in 4.3(c). Here "determine.module.function" is the procedure used to get the value for the FUNCTION attribute of the MODULE frame (fill in the FUNCTION slot of MODULE frame).

A class type relates those classes that have common attributes. "The shared attributes are then defined on the class type, and the class type is said to contain the classes" (Teknowledge, Inc., 1986, p. 2-6). For example, three types of course organization elements are defined as three classes: (a) MODULE, (b) INSTRUCTIONAL-MAP, and (c) INSTRUCTIONAL-MENU. The relational attributes such as CONSIST-OF and NEXT-ORGANIZATION-ELEMENT are shared by all of them. Thus, a class type called COURSE-ORGANIZATION-ELEMENT is created to contain the three elements.

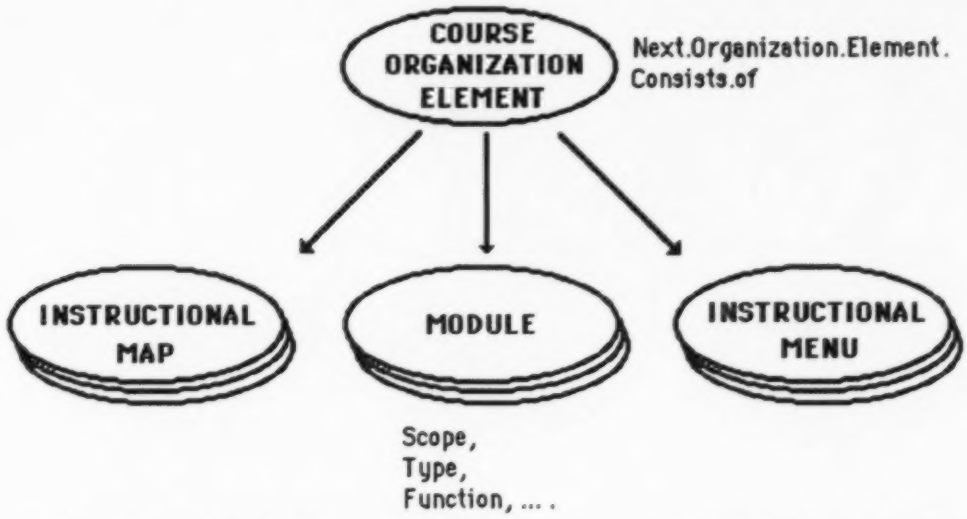


Figure 4.4. Relationships between Classes and Their Class Type.

The class types form a hierarchical concept structure or organization method for the domain, but do not themselves have instances. If attributes are defined at the class type level, any classes contained in the class type inherit all attribute definitions of the class type. If the attributes are defined at the class level, each instance inherits all attribute definitions of the class. As in Figure 4.4, NEXT-ORGANIZATION-ELEMENT and CONSISTS-OF attribute are shared by all three classes, while SCOPE, TYPE, and FUNCTION, etc. are only defined on the MODULE class.

Judgmental knowledge representation uses rules and certainty-factor objects. Rules in S.1 "can be expressed in English as a conditional or IF-THEN sentence" (Teknowledge, Inc., 1986, p. 2-7). The IF-part is called a premise, while the THEN-part is called conclusion as in Figure 4.5. If an attribute contains multiple values, multiple conclusions may appear in the CONCLUSION field of a rule definition as in Figure 4.5(b).

```
DEFINE RULE  REQUIRED.CONTENT.REPRESENTATION.001
::APPLIED.TO t:transaction
::PREMISE    transaction.name[t] is summary
::CONCLUSION required.content.representation[t] = definition
END.DEFINE
```

(a) Rule Definition with Single Conclusion.

```
DEFINE RULE  REQUIRED.CONTENT.REPRESENTATION.002
::APPLIED.TO t:transaction
::PREMISE    treansaction.name[t] is conversational.tutorial
::CONCLUSION begin
               required.content.representation[t] = definition;
               required.content.representation[t] = examples;
               required.content.representation[t] = context
             end
END.DEFINE
```

(b) Rule Definition with Multiple Conclusions.

Figure 4.5. Examples of Rule Definition in S.1.

S.1 uses a certainty factor mechanism introduced in the MYCIN system to represent the situation that facts and judgment may be believed with various degrees of certainty.

```
DEFINE CONTROL.BLOCK IDENTIFY.MODULES
::ARGUMENT   cs:content.structure,
             ch:coordddinate.set
::INVOCATION internal
::BODY
 begin
  invoke create.kinds.of.taxonomy(cs, ch);
  invoke dislay.kinds.of.taxonomy(cs);
  display "The following modules are required:" !
          new.line();
  if already.existing(root:coordinate.set |
    determined?(subordinate.set.of[root, ch]) and
    subordinate.set.of[root, ch])
  then invoke display.instructional.modules(root);
 end
END.DEFINE
```

Figure 4.6. An Example of Control Block.

Strategic knowledge is represented by control blocks and functions. S.1 contains a programming language similar to conventional languages such as PASCAL (Figure 4.6). Control blocks are used to implement the procedures that an expert uses in problem-solving. They specify "when to create class instances, when and how to conclude values for attributes, when to display the results to the user, and when to call other control blocks" (Teknowledge, Inc., 1986, p. 2-8).

Functions are used to compute a value used in attributes, rules, or control blocks. Both control blocks and functions can be defined, and associated with an attribute definition. Therefore, they are used to implement the procedural attachment aspect of a frame-based system. The difference is that functions are purely algorithmic in computing a value to fill in the slots of a frame, while control blocks can integrate algorithmic knowledge with judgmental knowledge in filling these slots.

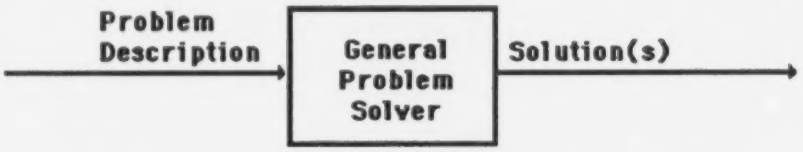
4.3. Reasoning Strategies

To solve an instructional design problem, an expert often uses different strategies at different times. This section illustrates various reasoning strategies implemented in the system.

Forward Reasoning

The ID Expert can be viewed as a problem solver in the area of instructional design. Figure 4.7(a) provides a "black box" definition of a General Purpose Problem Solver. Generally speaking, a problem solver takes a problem description as input, and reasons based on the data gathered from the problem description. Then it provides a solution to the given problem as output. Applying the same model to instructional

problems, we can define an Instructional Problem Solver as illustrated in Figure 4.7(b), where a problem description consists of audience information, and subject matter. The output consists of the course organization and related course specifications.



(a) A General Problem Solver.

(b) An Instructional Design Problem Solver.

Figure 4.7. Instructional Design Problem Solver.

One characteristic of solving an instructional design problem is that the complete solution would not be known unless the reasoning process is complete. Since there are no goal states known when the system begins to reason, the only inference mechanism available is forward reasoning. Harmon & King (1985) explain that:

> [A forward reasoning mechanism] is described as a "recognize-act" cycle. First, the rules that can succeed, given the contents of working memory, are recognized. One rule is selected, and then the action or conclusion is asserted into working memory. then the system proceeds to the next cycle and checks again to see what rules succeed. (p. 55)

Due to the nature of the inference engine in S.1 the forward reasoning control strategy was implemented in terms of control blocks and related procedural language. Depending on the situation, the strategy was explicitly or implicitly coded.

The explicit coding of a forward reasoning strategy involves uses the IF-THEN or CASE statements of S.1's programming language. Figure 4.8(a) presents, in plain English form, three rules for selecting which analytical

procedure to use in subject matter analysis. Figure 4.8(b) shows the control block implementation of these rules. Rule-002 and Rule-003 in figure 4.8(a) were combined in one IF-THEN-ELSE statement in figure 4.8(b). When the control block is invoked, the condition or premise part of each rule (the IF-part), is "recognized" in the order of their appearance in the control block. Once a rule succeeds the action defined in the THEN-part will be executed. Otherwise, either the ELSE-part or the next statement will be executed.

```
Rule-001: If content structure is 'kinds.of taxonomy'
          Then use concept taxonomy analysis.
Rule-002: If content structure is 'path'
          Then use procedural hierarchy analysis.
Rule-003: If content structure is not 'kinds.of' or 'path'
          Then exit consultation
```

(a) Rules for Selecting Analytical Methods.

```
          DEFINE CONTROL.BLOCK CONTENT.ANALYSIS
          ::ARGUMENT   c:course
          ::INVOCATION internal
          ::BODY
           begin
{Rule-001}   if content.structure[c] is "kinds of taxonomy"
             then invoke analyze.kinds.of.taxonomy(c);
{Rule-002}   if content.structure[c] is "path:parts"
             then invoke analyze.procedural.hierarchy(c)
{Rule-003}   else display "The design for " ! content.structure[c] !
                          " is out of current scope of the project."
           end
          END.DEFINE
```

(b). Procedural Implementation of Forward Reasoning Rules.

Figure 4.8. Use Control Block to Code Forward Reasoning Strategy.

The implicit representation of a forward reasoning strategy involves the repeated use of "invoke" statements to form a forward chain of actions. The forward reasoning strategy starts consultation from the initial state of the system and moves forward to the goal state for a solution. Since the searching space in seeking a solution for an instructional design problem is enormous, a space abstraction methods is used to simplify the problem. Stefik, et al. (1983) state that:

> Abstraction [of searching spaces] emphasizes the important considerations of a problem and enables its partitioning into subproblems. the simplest case is a fixed partitioning in the abstract space that is appropriate for all of the problems in an application. (p. 102)

The instructional design task performed by ID Expert can be viewed as a hierarchy of subtasks. Although many subtasks are performed iteratively in the conventional design process, the ID Expert prototype simplifies the process as the subtasks are performed in sequential order. ID Expert partitions the instructional design tasks into five ordered subtasks as follows:

1. Determine analytical method for a given subject matter and collect audience information.
2. Identify modules and determine instructional objectives for each module.
3. Organize modules into course organization diagrams.
4. Design modules including the functions and transactions.
5. Display integrated consultation report.

Each subtask listed above is further partitioned into sub-subtasks. For example "Identify module" for concept taxonomy contains following sub-tasks:

2.1. Create "kinds of" taxonomy.
2.2. Display "kinds of" taxonomy.
2.3. Identify integrative modules.
2.4. Identify instructional modules.
2.5. Display instructional objectives.

ID Expert defines each subtask as a control block, and uses the "invoke" statement to connect these control blocks forming a forward chain of reasoning. Figure 4.9 lists two control blocks for the subtasks identified here. Each invoke-statement can be represented by following statement:

If the current goal is to perform subtask A
Then invoke the control block that defines A.

Here, the IF-part of the rule is implicitly defined in the sequence associated with the appearance as the invoke-statement in the control block. It is always true when the statement is executed.

When a control block is invoked, the statements are executed in the same sequence as their appearance in the control block (except looping and branching). Once the premise of the first rule in the sequence is recognized, the action(s) defined in the THEN-part is activated. Then, this implementation will never require conflict resolution as in normal forward reasoning systems.

```
DEFINE CONTROL.BLOCK DESIGN.COURSE
::INVOCATION top.level
::BODY
 begin
   vars c:course;
   invoke display.welcome.message();
   create.instance course called c;
   invoke gather.audience.data(c);
   invoke identify.modules(c);
   create.instance course.organization called map;
   invoke organize.modules(c, map);
   invoke design.modules(c, map);
   invoke display.final.report(c)
 end
END.DEFINE
```

(a). Top Level Control Block.

```
DEFINE CONTROL.BLOCK IDENTIFY.MODULES
::INVOCATION c:course
::AUGUMENTS
::BODY
 begin
   vars struct.head:content.structure,
        coord.head: coordinate.set;
   create.instance content.structure called struct.head;
   create.instance coordinate.set called coord.head;
   invoke create.kinds.of.taxonomy(struct.head, coord.head);
   invoke display.kinds.of.taxonomy(struct.head);
   invoke identify.integrative.modules(struct.head, coord.head);
   invoke identify.instructional.modules(struct.head, coord.head);
   invoke display.instructional objectives(struct.head, coord.head);
 end
END.DEFINE
```

(b) Control Block for "Identify Modules" Procedure.

Figure 4.9. Forward Reasoning using Control Blocks and Invoke Statement.

Backward Reasoning

Although ID Expert uses forward reasoning as the main strategy in conducting instructional design activities, a backward reasoning strategy is also used to supplement and enhance the reasoning capabilities of the system.

Backward reasoning is goal-directed. Reasoning is a backward reasoning system starts at the goal state, and works "backward" through the subgoals to solve a problem. First, a rule that concludes about the current state is chosen to have its premise checked against the facts stored in the working memory. Its premise often contains attributes for which values are not known. As Harmon & King (1985) describe:

> [a backward reasoning strategy] identifies these new attributes as subgoals. Then other rules are examined that conclude values for this new subgoal attribute. Back-chaining [backward reasoning] continues until values are obtained for the subgoal attribute. Results are passed forward, and eventually conclusions are drawn about the original goal. (p. 54)

An instructional designer often involves selection tasks during a design process. That is, a category is selected from an established taxonomy. An instructional design selection task usually involves few categories. For example, Component Display Theory calls for a performance taxonomy with three categories: remember, use, and find, and a content taxonomy with four categories: fact, concept, procedure, and principle (Merrill, 1983). In designing a piece of instruction for a single idea, the designer must select appropriate categories from the performance, and the content taxonomy. The backward reasoning strategy is very efficient in solving this type of sub-problems because the possible outcomes are known, and they are a reasonable small in number (Harmon & King, 1985).

For example, ID Expert can design instructions on both concept taxonomy and procedural hierarchy structure. An important decision must be made at the initial stage of the consultation is which analytical method should the system use to analyze the subject matter. Thus, the first goal of the system is to determine the content structure category that the subject matter belongs. The task is to select a category from a list of possible structures presented by the system. Since S1.'s inference-engine uses backward reasoning as its default control strategy, the implementation involves representing the reasoning process through judgmental rules. Figure 4.10 shows the inference process that concludes content structure to

be "path:parts" (a form of procedural hierarchy). During the reasoning many rules may have been tried but only those that succeed are shown in the diagram. The numbers in circles indicates the order in which events occur during the consultation.

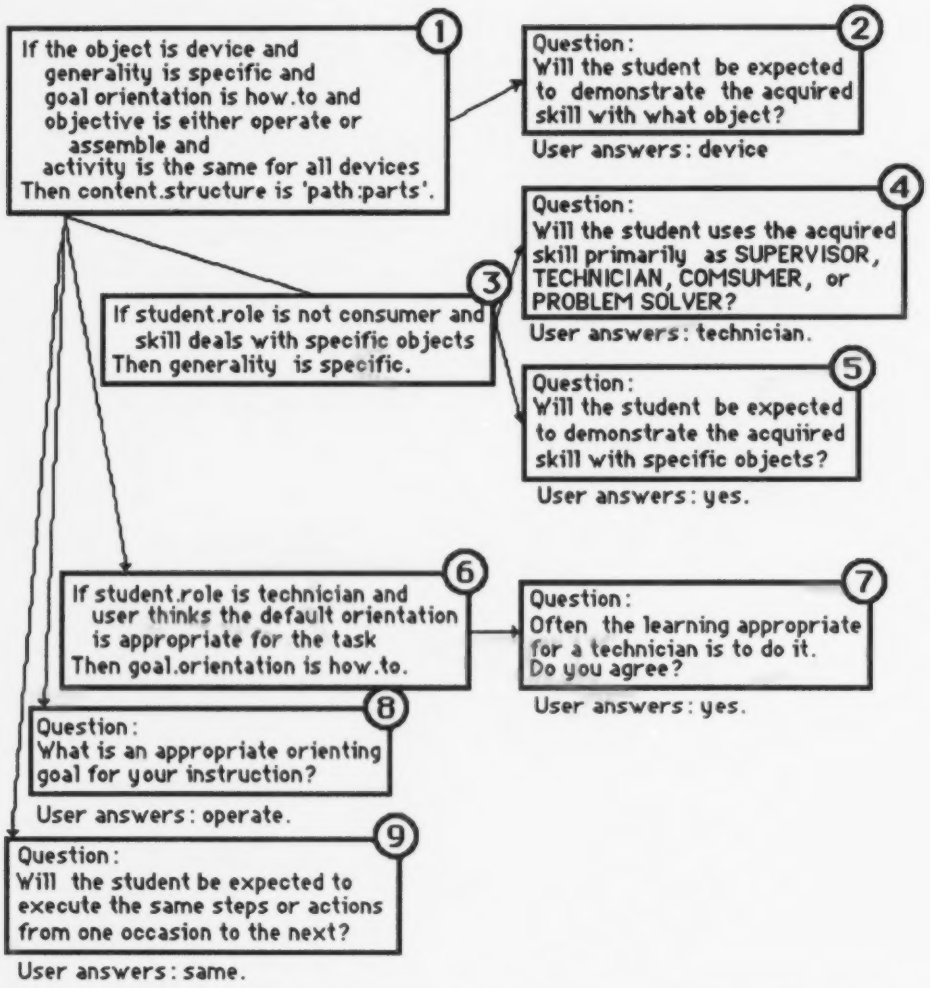


Figure 4.10. An Inference Tree Generated by Backward Reasoning Strategy.

Task-specific Reasoning Control

True expertise in problem-solving means knowing not only what information to collect but also when and how to collect it. When a system generates questions to get information, the order in which the questions are asked also reflects the reasoning strategies used. The task-specific reasoning control strategy is used to implement this type of reasoning behavior.

The task-specific reasoning control strategy deals with the control of the determination of attribute values. It offers control over when, what, and how to get information during a consultation. It does not separate

itself from other reasoning strategies such as backward and forward reasoning. Rather, it often combines those strategies for explicit control of an expert system's reasoning behavior.

"When to" refers to the order in which data is collected. For user-acceptance of the expert system, the reasoning style of the system should be as close to that of the expert as possible. ID Expert organizes data collection in a manner natural to an instructional design process rather than bounded by the default order (backward chaining) of the S.1 inference-engine. Data with similar properties are grouped together, and some of them may even be used later in the consultation. For example, audience data such as LOCUS-OF-CONTROL, and MOTIVATION are collected at the beginning of a consultation, although the information will be used later in the DESIGN-MODULE phase. The same is true for the information associated with a concept taxonomy. Information about a coordinate set such as COORDINATE-SET-RELATIONSHIP, COORDINATE-SET-FOCUS, and the characteristics of concepts in the coordinate set, are asked as soon as the coordinate set is created although they do not facilitate the creation of a concept taxonomy tree structure.

"What to" refers to the control over asking the right questions at the right time. During a consultation ID Expert frequently asks a user to select a category from a taxonomy. For a novice user, selecting from a taxonomy may not be a trivial task. More guidance is necessary to help the user make decisions. However, if the user is familiar with the taxonomy, or has been gaining experience with the system at later stages of a consultation, seeking information using guidance may be redundant and boring. To provide the guidance to the first group of users, and also to provide the flexibility for the second group of users, ID Expert divides all questions into two levels. The first level is always a direct question asking the user to supply attribute value. If a user answers "unknown", ID Expert takes that as a hint and assumes that the user may have trouble in selecting a value from the category. Therefore, a more detailed question series is generated to assist the user in decision-making. The portions of a transcript presented in Figure 4.11 illustrate how ID Expert interacts with a user in case the user does not know the value for the STUDENT-MOTIVATION attribute.

A simple implementation of the reasoning strategy can be explicitly expressed as:

Seek a[t] by asking;

If a[t] not.known

Then seek a[t] by rules;

Here a[t] means the attribute (a) of an object (t). The strategy first asks the user to select a category from a list of possible choices. If the user indicates he/she does not know the results by entering "unknown", then ID Expert tries those rules that may conclude the value of the attribute. Usually, the rules work in a backward reasoning strategy as defaulted in S.1's inference-engine. The backward reasoning initiates a set of more detailed questions to determine the value for the attribute. This reasoning strategy has been implemented on attributes like COORDINATE-SET-RELATIONSHIP, COORDINATE-SET-FOCUS, and STUDENT-MOTIVATION, etc. Each of them has a control block attached. This type of control block is specifically defined as a determination control block, which represents the task-specific control strategy used by ID Expert to determine the attribute value.

4: What is the average level of motivation for this group of students?
4> UNKNOWN

{IDS initiates a more detailed set of questions that enable the system to recommend the level of student motivation.}

5: Why will the students take this instruction?
Volunteer - Students will have chosen to take this instruction.
Required - Students will be required to take this instruction.
5> REQUIRED

6: Will this instruction contribute to a job change for the students?
6> YES

7: Will this instruction enables or requires a job change for the students?
7> YES

8: Did the students request the job change which requires this instruction?
8>REQUESTED

INTERMEDIATE RESULT:
It has been determined that the level of motivation for this group of students in HIGH.

Figure 4.11. A Portion of a Dialogue with ID Expert to Determine the Level of Motivation.

The "how to" refers to the means that ID Expert uses to acquire values for an attributes. There are four basic means to determine values the the attribute: by asking user, by using rules, by transferring from attributes of other instances in the same class, and by executing the procedure or function attached to the attribute. Asking user and using rules are obvious means, and need no further explanation. Transferring attribute values is also referred to as knowledge propagation. The value of an attribute determined at one stage of consultation may be transferred to other attributes at a later stage. For example, student familiarization is an attribute defined on module class to indicate the degree that the student familiarity with the content of instruction. The information is crucial in selecting functions that should be performed by a module. Since a preintegrative and postintegrative module pair contain the same content, ID Expert only asks the student familiarization question when determining the module function for a preintegrative module. It then transfers (propagates) the value to its corresponding postintegrative module. A more sophisticated control strategy requires a combination of other methods to determine an attribute value. The example used in explaining "what to" is a combination of methods (asking user, and using rules), which was implemented as a control block.

Reasoning with Certainty Factor

An IF-THEN rule can be considered as a causal model. That is, if A then B leads to the conclusion that if A is true, then B must be true. This does not apply well when making instructional design decisions. Rather, the model that is commonly used for instructional design, as well as in other social sciences involves statements like: if A, then B may be true. In other words, the judgmental knowledge of instructional design usually involves uncertainty rules. Certainty factors are used to represent these uncertainty rules. Figure 4.12 presents several example of uncertainty rules that accumulate evidence that the transaction should be SUMMARY.

Rule-001 If module function is overview
Then transaction type is summary <0.40>

Rule-002 If module function is new presentation
Then transaction type is summary <-1.00>

Rule-011 If coordinate concept relationship is independent
Then transaction type is summary <0.10>

Rule-021 If concept characteistics is abstract
Then transaction type is summary <-0.10>

Figure 4.12. Rules with Certainty Factors.

The value of confidence ranges from -1 to 1. If the current goal in a consultation is to determine values for the TRANSACTION attribute, all rules that conclude with that attribute will be tried. Each rule that succeeds adds the evidence to the working memory using certainty factor calculus (Rich, 1983). A rule that concludes with a positive certainty factor (e.g. Rule-001) increases the certainty factor accumulated, and a rule that concludes with a negative certainty factor (e.g. Rule-021) decreases the certainty factor accumulated. Once an attribute value reaches a number of -1 or 1, the system stops examining the rules concluding the same value. Therefore, the rules that conclude a value with certainty factor of -1 or 1 (e.g. Rule-002 in Figure 4.12) are more decisive than others. ID Expert uses this property to increase the efficiency of reasoning. For example, transaction type requires more than one hundred rules providing evidence accumulation. A crucial factor that contributes to TRANSACTION determination is MODULE-FUNCTION, since it contains a large portion of decisive rules. ID Expert divides all rules into two categories. The first category contains rules that use the MODULE-FUNCTION attribute only to make a conclusion, while the second category contains the rest of the rules. During a consultation, the rules in the first category will be tried first. In most cases , the success of a rule in the first category will eliminate a large portion of the rules in the second category. For example, if the result MODULE-FUNCTION is obtained to be OVERVIEW, after trying the rules in the first category, the working memory will contain the following entries for the TRANSACTION attributes:

synthesis <0.40>
summary <0.30>
exposition <0.10>
conversational tutorial <0.20>
compare and contrast <0.20>,

edit <-1.0>
interpret <-1.0>
sort <-1.0>
label <-1.0>
rank <-1.0>
question <0.10>

Since EDIT, INTERPRET, SORT, LABEL, and RANK all have a -1.0 certainty factor, only those rules that do not conclude these values will be tried. On average this strategy saves half of the search time usually required.

S.1 provides a mechanism for translating certainty factor into English sentences. This mechanism is utilized in ID Expert when presenting conclusions with certainty factor. Figure 4.13 presents a transcript that illustrates the result after accumulation of evidence using certainty factor.

```
{After user select assessment.diagnosis as module function,
 IDS tries to accumulate evidence for transaction type
 appropriate for the module function, and the results are:}
Recommended transactions for assessment.diagnosis are:

 There is strong evidence <.82> that transaction should be sort.
 There is suggestive evidence <.62> that transaction should be label.
 There is weakly suggestive evidence <.36> that transaction should be edit.

53: Which transaction would you select?
53> SORT
```

Figure 4.13. Sample Output of Evidence Accumulation Using Certainty Factor.

Nonmonotonic Reasoning

The inference-engine in S.1 is monotonic in nature. That is, the "attributes determined for a given instance of a class cannot be redetermined for that instance" (Teknowledge, Inc., 1987. p. 3-8). when a new fact is added to the system during consultation, no checks are made to see if there are inconsistencies between the new fact and the old knowledge (Rich, 1983). Therefore, S.! is not very good at dealing with the kinds of situations that often arise in real problem domains, i.e. incomplete information, and reconsideration.

During an instructional design consultation, few expert instructional designers are able to collect all the information necessary to make the design decisions. When such information is lacking, the expert often makes sensible guesses, as long as no contradictory evidence is present. Rich (1983) states that

> The construction of these guesses is known as default reasoning. ...
>
> This sort of default-reasoning is nonmonotonic (i.e., the addition of one piece of instruction may force the deletion of another) because statements that are derived in this way depend on the lack of belief in certain other statements. This means that if one of those previously lacking statements is added to the system, the default-generated statements will have to be deleted. (p. 176)

Due to this limitation of S.1, it is impossible to implement a full default-reasoning system as described. ID Expert uses default values where required information is lacking. the default values are specified statically or dynamically. A static specification inserts the default values in the DEFAULT.VALUE field of an attribute definition. In ID Expert the defaults for all relational attributes is the value of FALSE. The default statement is deleted if the attributes are determined as TRUE during a consultation.

An expert can usually specify default values used during a consultation from prior experience. But, such value may also be obtained from empirical research. For example the default values for the STUDENT-MOTIVATION attribute can be empirical obtained. Assuming the score of target population forms a normal distribution, the ID Expert default for the STUDENT-MOTIVATION is a value set of [low <.20>, moderate <.60>, high <.20>].

```
DEFINE RULE  COORD.SET.FOCUS.CRD01
::APPLIED.TO  c:coordinate.set.type
::PREMISE     discriminate.among.concepts[c]
::CONCLUSION designer.selected.focus[c] = discrimination
::CATEGORIES {derived}
END.DEFINE

DEFINE RULE  COORD.SET.FOCUS.DEFAULT.CRD
::APPLIED.TO  c:coordinate.set.type
::PREMISE     coord.set.relationship[c] is in coordinate
::CONCLUSION designer.selected.focus[c] = discrimination
::CATEGORIES {default}
END.DEFINE

DEFINE CONTROL.BLOCK DETERMINE.FOCUS
::ARGUMENT  a:attribute,c:coordinate.set.type
::INVOCATION determination
::BODY
 begin
   seek a[c] by asking;
   if a[c] not.known then
   begin
     seek a[c] by rules whose.category is drived;
     if a[c] not.known then
       seek a[c] by rules whose.category is default;
   end
END.DEFINE
```

Figure 4.14. Dynamic Specification of Defaults.

In most situations, the default value can not be pre-determined. Rather, the determination of the default value requires other information which is only available during a consultation. ID Expert uses judgmental rules plus a task-specific control strategy to implement the dynamic specification of default value. Figure 4.14 illustrates the implementation of dynamic specification of default values.

Here two rule categories are defined. The first category specifies the rules that may derive a value from direct questioning the user for the value of the COORDINATE-SET-FOCUS attribute. The second set of rules default the value based on the information about the COORDINATE-SET-RELATIONSHIP attribute. The DETERMINE.FOCUS control block specifies the task-specific control strategy for determining the value of COORDINATE-SET-FOCUS. The system first asks user about the value of COORDINATE-SET-FOCUS. If the user answers "unknown", the system tries to derive the value by invoking the rules in the "derived" category. If the system is still unable to determine the value, then the rules in the "default" category are tried. In Figure 4.14, if the information on the COORDINATE-SET-RELATIONSHIP attribute is COORDINATE, then the default value for COORDINATE-SET-FOCUS becomes DISCRIMINATION.

Another way to specify a default value is through the accumulation of certainty factors. The default is to select a value with highest certainty factor. Figure 4.15 demonstrates this capability through a transcript from a consultation with ID Expert.

{After user select assessment.diagnosis as module function, IDS tries to accumulate evidence for transaction type appropriate for the module function, and the results are:}

Recommended transactions for assessment.diagnosis are:

There is strong evidence <.82> that transaction should be sort.
There is suggestive evidence <.62> that transaction should be label.
There is weakly suggestive evidence <.36> that transaction should be edit.

53: Which transaction would you select?
53> UNKNOWN

I select SORT as the transaction for assessment.diagnosis function.

{When user answers 'UNKNOWN', IDS select the transaction with highest certainty value.}

Figure 4.15. Selecting Highest Certainty Value as Defaults.

To solve the limitation posed by the monotonic nature of S.1, ID Expert first tries all other means such as asking the suer, trying other rules, to determine a value fro an attribute before trying the rules that produce the defaults. Once the attribute of an instance uses the default values, the system continue to reason using the default values, and no attempts are made to modify the attribute. Though not considered as tree nonmonotonic reasoning, this implementation allows ID Expert to continue the consultation with lack of certain information.

Reconsideration is another behavior demonstrated by experts during an instructional design consultation, and does not have straight forward implementation in S.1. In a design consultation, an expert needs to check certain values of critical attributes before moving on. To check a value, the system must reason with the value. If the value is found unacceptable, it must be redetermined, which is not permitted by the monotonic inference-engine of S.1.

To simulate the reconsideration behavior, a class type was created to contain a BUFFER class along with the class that has attributes to be

reconsidered. All those attributes are defined on the class type. To determine an attribute with reconsideration, ID Expert crates an instance of the BUFFER class, and determines the value of the attribute of the instance. Then, if the value is found to be acceptable, it is then copied to the original attribute using a TRANSFER function defined in the system. Otherwise, a new instance of the BUFFER type is created and checked again. The process is repeated until an acceptable value is found. Figure 4.16 illustrates the reconsideration process using COORDINATE-SET-RELATIONSHIP, and COORDINATE-SET-FOCUS attributes as an example.

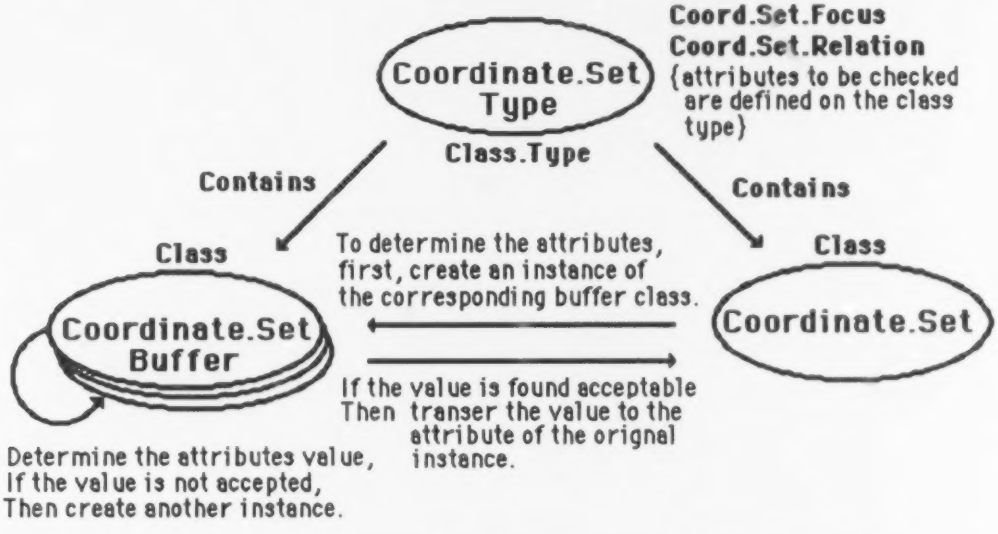


Figure 4.16. Reconsideration Process in ID Expert.

Each time an instance of coordinate set is created, ID Expert queries the value for COORDINATE-SET-RELATIONSHIP and COORDINATE-SET-FOCUS information. Since the information is vital to other decision making, it must be sure that the two values are consistent with each other. As an example, selecting a DISCRIMINATION focus is inconsistent if the value of COORDINATE-SET-RELATIONSHIP is INDEPENDENT. In the transcript presented in Figure 4.17, the user's belief of COORDINATE-SET-RELATIONSHIP is modified after ID Expert points ou;the inconsistency between the COORDINATE-SET-RELATIONSHIP, and COORDINATE-SET-FOCUS specified.

24: Do you wish to expand on concept: carnivore?
24> YES

25: What is the coordinate set relation for subordinate concepts of carnivore?
25> INDEPENDENT

26: What are the subordinate concepts of carnivore?
26> SEEBIRD.CARNIVORE, HUNTER.CARNIVORE

{Here, the user made a mistake in specifying coordinate set relationhsip: it should be coordinate rather than independent.}

27: What is the primary focus to learn?
27> DISCRIMINATE

{After the user indicated that the focus is discrimination, IDS detected the inconsistancy and generated appropriate message.}

28: Since it is unlikely taht students will confuse instances of SEEBIRD.CARNIVORE and HUNTER.CARNIVORE then it is usually not necessary for student to learn to discriminate instances of SEEBIRD.CARNIVORE and HUNTER.CARNIVORE from one another. Would you like to change the information on coordinate set focus or relation?
28> YES

29: What information about the coordinate set would you like to change?
29> RELATION

{After the user indicated that coordinate relationship needs to be changed, a new instance is created internally and the value of coordinate set relation is re-determined.}

30: What is the coordinate set relationship?
30> COORDINATE

Figure 4.17. A Transcript of Reconsideration dialogue.

4.4. User Interface

This section describes some key issues in user interface implementation. Some aspects of user interface design have also been discussed in a companion report bu Merrill and Li (1988).

Text Representation of Graphic Output

An instructional design specification for a course usually contains graphic representation of content structure and course structure. However, the hardware environment in that ID Expert was developed does not have graphics capability. As a prototype system, ID Expert uses a text representation instead.

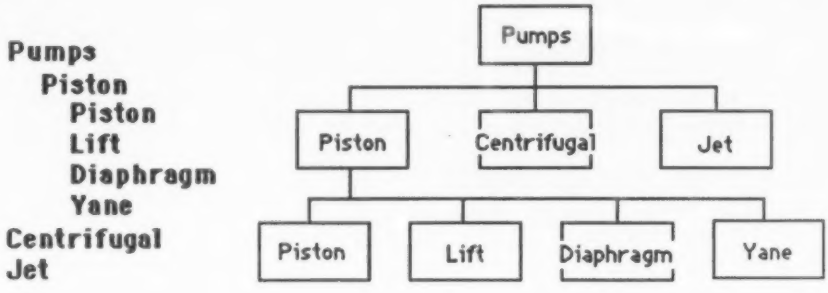


Figure 4.18. Text Presentation of Content Structure.

Content structure of taxonomy or hierarchy types can be clearly represented in terms of outline form. the concepts at the same level of the taxonomy start at the same column in the outline form as in Figure 4.18.

A text representation of course organization is also straight forward. It contains four statements each describing the preintegrative module, first instructional menu, second instructional menu, and postintegrative module respectively as in Figure 4.19.

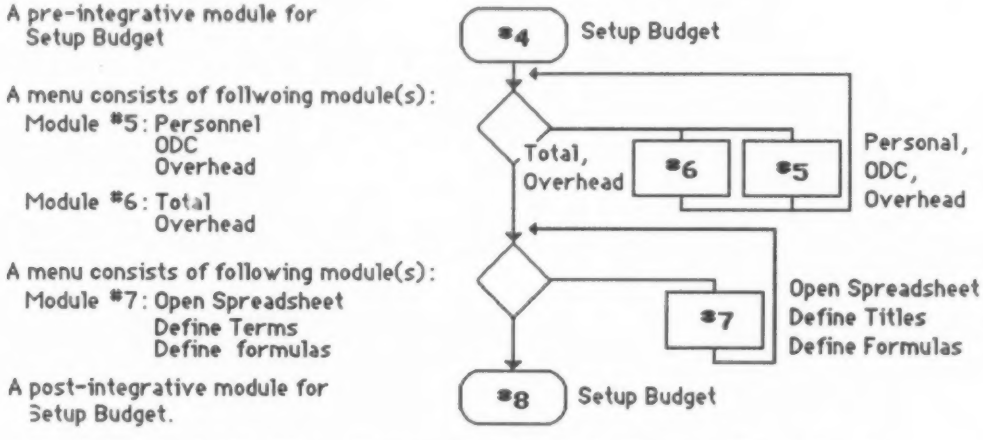


Figure 4.19. Text Representation of Course Organization.

Help Facilities

During an instructional design consultation, ID Expert uses many taxonomies in reasoning. Some of them do not have a commonly agreed-on definition among instructional design theorists and practitioners. The others may have just been created during the system development. Thus, when ID Expert asks a user to select a category from a list possible choices, the user may get confused and have difficulty in making the selection. A HELP facility is implemented in ID Expert to provide more detailed information such as the definition of terms, and examples for each category. The HELP messages will be displayed when the user types "HLEP" to request it or the user makes a mistake in responding to system questions. Figure 4.20 is a transcript that demonstrates the help message presented by ID Expert.

2: What is the locus of control of this group of students?
Type HELP for more information.
2>HELP
Locus of control is a variable which refers to the amount of control students feel that they have in a situation. When students feel that they are in control they are willing to make decisions and to direct their own activities. When students feel that others are in control they are more willing to let others direct them and make decisions for them. If you have data on the general orientation of the students either from formal tests of locus of control or from observing the students, this information is helpful in making instructional decisions. If the students are managers, supervisors or others leaders their orientation is likely to be INTERNAL. If the students are not in a leadership position then their orientation is often EXTERNAL.

EXTERNAL locus of control means that the students generally feel that others are in control.

INTERNAL locus of control means that the students generally feel that they are in conntrol.

Figure 4.20. A Help Message Generated by ID Expert, which Includes Examples, and Definitions of Key Terms.

4.5. Test and Evaluation

Gaschnig, Klahr, Pople, Shortliffe, and Terry (1983) state that:

> The [expert system] evaluation process should be continual, beginning with system design, extending in an informal fashion through the early stages of development, and becoming increasingly formal as a developing expert system begins to achieve real-world implementation. (p. 257)

Since this project focused on the development of a prototype instructional design expert system, no formal evaluation studies were conducted. However, informal evaluations are frequently performed during the system development. This section describes the informal evaluation process of the system.

Aspects of ID Expert Being Evaluated

During an expert system development process, which aspect of the system should be evaluated depends on task, system, goals, and stages of development. Gaschnig, et al. (1983) suggest that five aspects of an expert system should be included in an evaluation study of the system. These

five aspects are (a) the quality of the system's decision-making and advice; (b) the correctness of the reasoning methods used; (c) the quality of the user-interface; (d) the system's efficiency; and (e) the system's cost-effectiveness.

However, ID Expert is a prototype instructional design system. A prototype system only implements some aspects of a "real-world" system. Gehani (1983) states that

> the design goal of a prototype are quite different from those of the final product. A prototype system is constructed to illustrate the feasibility of new ideas or design. Unlike the final product, prototype design and construction often begin with incomplete specifications. Mistakes in its design are remedied by quick and local modifications. Short cuts are taken to produce the prototype quickly at the expense of efficiency, robustness, generality and a good user interface. Available (not the best) tools and systems are used to minimize cost and time. Consequently, a prototype system should be considered to be a throw away system and the final system build from scratch. (p. 480)

Therefore, the evaluation of a prototype can only focus on a subset of the five aspects of expert system evaluation just mentioned. We emphasize the evaluation of the quality of the system's decision-making, and the correctness of the reasoning strategies used. Since different decision-makings may use different reasoning strategies, both aspects were evaluated together.

Methods of Evaluation

Exemplary cases play an important role in expert system development and evaluation. They are frequently used by knowledge engineers and domain experts in all phases of the knowledge engineering process. There were two kinds of exemplary cases used in the evaluation of ID Expert: (a) those used by the knowledge engineer in testing and debugging the prototype system; and (b) those used by the domain expert in evaluating the system as a whole. The former corresponds to a formative evaluation, while the later corresponds to a summative evaluation, although both are conducted informally.

Gaschnig, et al. (1983) describe how evaluations are useful in system development:

> evaluations pervade the system-building process and are crucial for improving system design and performance. Every time a rule in the knowledge base is changed, added, or deleted, every time the code of the reasoning program is modified or extended, or every time the knowledge representation scheme is refined, action has been taken in response to an informal evaluation. (p. 242)

ID Expert consists of twenty-three additive knowledge base (see Table 4.1). Each additive knowledge base implemented no more than three key decisions or advice of the system. During the ID Expert development, each of the knowledge base constituted an evaluation unit. Test cases were prepared to evaluate the decision-making for each additive knowledge base. For each additive knowledge base, the evaluations were performed in two stages. First, the test cases used to evaluate the previous knowledge base were run through the system to ensure that the addition of new knowledge will not change the established decision-making of the system. Second, either new cases were prepared by the domain expert to evaluate the knowledge base as a whole, or previous cases were expanded to test the new decision-making mechanism added. These tests made sure that once instructional design knowledge was elicited from the domain expert it was correctly implemented in the system.

The domain expert also involved in the evaluation process regards to the embedded domain knowledge and how it is used by the system.

> The experts repeatedly perform both static and dynamic evaluations. In static evaluation they compare the system's knowledge base with their own, looking for consistency and completeness. In dynamic evaluation they compare the system's line of reasoning and conclusions on a specific case with their own. (Gaschnig, et al. 1983, p. 245)

In a prototype system, the design emphasis is the knowledge representation, and reasoning strategies suitable for a given domain. We

focus on the dynamic evaluation from the domain expert because the most important goal is to demonstrate the feasibility of knowledge engineering approach to instructional design problems. In ID Expert development, a preselected case was used during the development. When the system was functional, the domain expert was requested to generate more cases to test the reasoning capability of the system.

Reference

Aronson, D. T., & Briggs, L. J. (1983). Contributions of Gagne and Briggs to a prescriptive model of instruction. In C. M. Reigeluth (Ed.), Instructional design theories and models: an overview of their current status, (pp. 75-100). Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers.

Baker, R. L., & Gerlach, V. S. (1971). Constructing objectives of cognitive behavior. In R. L. Baker, & R. E. Schutz (Eds.), Instructional product development, (pp. 25-64). New York: Van Nostrand Reinhold Company.

Briggs, L. J. (1977). Organizing the course. In L. J. Briggs (Ed.), Instructional design -- principles and applications (pp. 99-112). Englewood Cliffs, NJ: Educational technology publications.

Buchanan, B. G., Barstow, D., Bechtel, R., Bennett, J., Clancey, W., Kulikowski, C., Mitchell, T., & Waterman, D. A. (1983). Constructing an expert system. In R. Hayes-Roth, D. A. Waterman, & D. B. Lenat (Eds.), Building expert systems. Reading, MA: Addison-Wesley Publishing Company, Inc.

Donald, J. G. (1987). Learning schemata: methods of representing congnitive, content and curriculum structures in higher education. Instructional Science, 16, 187-211.

Donald, J. G. (1983). Knowledge structures: Methods for exploring course content. Journal of Higher Education, 54(1), 31-41.

DuBois, F. N., Alverson, G. F., & Staley, R. K. (1979). Educational psychology and instructional decisions. Homewood, IL: The Dorsey Press.

Gagne, R. M., & Briggs, L. J. (1974). Principles of instructional design. New York: Holt, Rinehart, and Winston, Inc.

Gaschnig, J., Klahr, P., Pople, H., Shortliffe, E., & Terry, A. (1983). Evaluation of expert systems: issues and case studies. In R. Hayes-Roth, D. A. Waterman, & D. B. Lenat (Eds.), Building expert systems. Reading, MA: Addison-Wesley Publishing Company, Inc.

Gehani, N, H, (1981). An electronic form system-an experience in prototyping. Software--Practice and Experience, 13, 479-486.

Harmon, P., & King, D. (1985). Expert systems. New York: John Wiley & Sons, Inc.

Hayes-Roth, F., Waterman, D. A., & Lenat, D. B. (Eds.) (1983). Building expert systems. Reading, MA: Addison-Wesley Publishing Company, Inc.

Horowitz, E., & Williamson, R. (1984). SODOS a software documentation support environment: its definition. Unpublished manuscript, Univeristy of Southern California, Computer Science Department, Los Angeles.

Human Technology, Inc. (1986). An expert system for instructional design: results of problem assessment and design for prototype development. McLean, VA: Author.

Li, Z. (1988). IDS: A prototype instructional design expert system. Dissertation, Univeristy of Southern California, Los Angeles.

Merrill, M. D. (1987). An expert system for instructional design. IEEE Expert, summer, 25-37.

Merrill, M. D. (1983). The component display theory. In C. M. Reigeluth (Ed.), Instructional-design theories and models: an overview of their current status, (pp. 279-333). Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers.

Merrill, M. D., & Li, Z. (1988). Implementation of an expert system for instructional design: phase 2. Alexandria, VA: Army Research Institute for the Behavioral and Social Sciences.

Merrill, M. D., & Tennyson, R. D. (1977). Teaching concepts: an instructional design guide. Englewood Cliffs, NJ: Educational Technology Publications.

Rich, E. (1983). Artificial Intellignece. New York: McGraw-Hill Book Company.

Reigeluth, C. M. (1983). Instructional design theories and models: an overview of their current status. Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers.

Reigeluth, C. M., & Merrill, M. D. (1984). Extended Task Analysis Procedure (ETAP). New York: University Press.

Reigeluth, C. M., Merrill, M. D., & Bunderson, C. V. (1978). The structure of subject matter content and its instructional design implications. Instructional Science, 7, 107-126.

Reigeluth, C. M., & Stein, F. S. (1983). The elaboration theory of instruction. In C. M. Reigeluth (ed.), Instructional deisgn theories and models: an overview of their current status, (pp. 335-381). Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers.

Romiszowshi, A. J. (1981). Designing instructional systems: decision making in course planning and curriculum design. New York: Nichols Publishing.

Teknowledge, Inc. (1987). S.1 advanced skill training materials. Palo Alto, CA: Author.

Teknowledge, Inc. (1986). S.1 reference manual, version 2.2. Palo Alto, CA: Author.

Von Pattern, J., Chao, C., & Reigeluth, C. M. (1986). A review of strategies for sequencing and synthesizing instruction. Review of Educational Research, 56(4), 437-471.

Waterman, D. A. (1985). A guide to expert systems. Reading, MA: Addison-Wesley Publishing Company, Inc.

Winston, P. H. (1984). Perspective. In P. H. Winston, & K. A. Prendergast (Eds.), The AI business -- the commercial uses of artificial intelligence. Cambridge, MA: The MIT Press.